Appendix: Virtual Occupancy Sensing: Using Smart Meters to Indicate Your Presence

Ming Jin, *Member, IEEE,* Ruoxi Jia, *Member, IEEE,* and Costas J. Spanos, *Fellow, IEEE*

Appendix A
Multiview-based Iteration Training

Assign the first view, $\mathcal{X}_{v1}$, as the time of the day, and the second view, $\mathcal{X}_{v2}$, as the power-derived features, and let f_{v1} and f_{v2} denote the classifiers based on the two views respectively. Given the unlabeled dataset $S = \{(\boldsymbol{x}_1^{v1}, \boldsymbol{x}_1^{v2}), \cdots, (\boldsymbol{x}_n^{v1}, \boldsymbol{x}_n^{v2})\}$, where $\boldsymbol{x}_i^{v1} \in \mathcal{X}_{v1}$ and $\boldsymbol{x}_i^{v2} \in \mathcal{X}_{v2}$, MIT proceeds as follows:

1) *Initialization:* Set the initial training set $L^1 = \{(\boldsymbol{x}_1^{v1}, \boldsymbol{x}_1^{v2}, \hat{y}_1), \cdots, (\boldsymbol{x}_n^{v1}, \boldsymbol{x}_n^{v2}, \hat{y}_n)\}$, where $\hat{y}_i = f_{v1}(\boldsymbol{x}_i^{v1})$ according to the prior information provided by the time view, i.e. (rough) occupancy schedules (line 1 in Algorithm 1).
2) *Multiview training:* For rounds $t = 1, 2, ...$, train the power-based classifiers with L^t to obtain $L^t_{new} = \{(\boldsymbol{x}_1^{v1}, \boldsymbol{x}_1^{v2}, \tilde{y}_1), \cdots, (\boldsymbol{x}_n^{v1}, \boldsymbol{x}_n^{v2}, \tilde{y}_n)\}$, where $\tilde{y}_i$ is the majority votes among the power- and time-based classifiers (lines 2 to 7).
3) *Labeled set updates:* In the next round,
$$L_j^{t+1} = \{L_j^t \cap L_{j,n}^t\} \cup Sample\{L_j^t \Delta L_{j,n}^t; \alpha_j\} \quad (1)$$
for $j \in \{-1, +1\}$, where $L_j^t = \{((\boldsymbol{x}_i^{v1}, \boldsymbol{x}_i^{v2}, \hat{y}_i) | \hat{y}_i = j\}$, and $L_{j,n}^t$ denote the set of *new* samples whose labels are j, $L_j^t \Delta L_{j,n}^t$ is the symmetric difference set operation, and α_j is the sampling rate for label $j \in \{-1, +1\}$ (line 8).
4) *Stopping condition:* stop the iteration whenever (7) in Theorem 3.3 is satisfied (line 9).

Algorithm 1: Pseudo-code of Multiview Iteration

```
Multiview_Iteration($X$, Prior, MaxIter)
Input: $X$: Feature matrix of size $n \times m$, where $n$ is the
       number of samples, $m$ is the number of views.
       Prior: Expert knowledge for initialization.
       MaxIter: Maximum iteration number.
Initialization:
1  $L^1_{+1}, L^1_{-1} \leftarrow$ Prior($X$)      // initial estimation by Prior
   stopCond $\leftarrow false$                  // stop condition
   $t \leftarrow 1$                              // iteration number
   $\eta_1 \leftarrow 0.5$                        // training noise rate
Main Program:
while $\neg$stopCond $\wedge$ $t <$ MaxIter do
    YMat $\leftarrow$ emptyMatrix($n, m$)
2   for $v_{ind} \in \{1, ..., \nu\}$ do
3       EstMdl $\leftarrow$ MdlEst($X(\cdot, v_{ind}), L^t_{+1}, L^t_{-1}$)
4       YMat($\cdot, v_{ind}$) $\leftarrow$ MdlPred(EstMdl, $X(\cdot, v_{ind})$)
5   for $s_{ind} \in \{1, ..., n\}$ do
6       $Y(s_{ind}) \leftarrow$ MajVote(YMat($s_{ind}, \cdot$))
7   $L^t_{j,n} \leftarrow$ getSet($Y$), $j \in \{-1, +1\}$
8   $L^{t+1}_j = \{L^t_j \cap L^t_{j,n}\} \cup$ Sample$\{L^t_j \Delta L^t_{j,n}; \alpha_j\}$,
    $j \in \{-1, +1\}$
    $\eta_{t+1} \leftarrow$ EstEta($L^t_{-1,+1}, L^{t+1}_{-1,+1}, \alpha_{-1}, \alpha_{+1}, V^{est}$)($*$)
9   stopCond $\leftarrow$ checkStop($L^t_{-1,+1}, L^{t+1}_{-1,+1}, \eta_{t,t+1}$)
    $t \leftarrow t + 1$
Output: $L_{-1}, L_{+1} \leftarrow L^t_{-1}, L^t_{+1}$   // labeled datasets
```

Appendix B
Derivation of the Surrogate Loss

The basic learning problem (BL) is described by $(l, \mathcal{F}, e_n)$, where $l : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$ is the *loss function* to penalize misdetection, $\mathcal{F}$ is the class of classifiers, $e_n : \mathbb{D} \to (\mathcal{X}, \mathcal{Y})^n$ is the repetitive experiments performed to acquire the dataset, $S = \{(\boldsymbol{x}_1, y_1), \cdots, (\boldsymbol{x}_n, y_n)\} \sim e_n(\mathbb{D})$, and $\mathbb{D}$ is the distribution between power and occupancy,

To motivate the **learning under corruption scheme**, we introduce the *corruption process* $T : \mathcal{O} \to \tilde{\mathcal{O}}$ as a Markov kernel, which corrupts the outcome $\mathcal{O}$ of the experiments to be $\tilde{\mathcal{O}}$, i.e., $\tilde{e}_n = T(e_n)$. Each Markov kernel is associated with a linear mapping, $T : (\mathbb{R}^{\mathcal{O}})^* \to (\mathbb{R}^{\tilde{\mathcal{O}}})^*$, where $(\mathbb{R}^{\mathcal{O}})^*$ is the dual space of $(\mathbb{R}^{\mathcal{O}})$ for linear functionals. The learning problem is characterized by $(l, \mathcal{F}, \tilde{e}_n)$, as compared to the original BL problem.

Definition B.1 (Reconstructible Markov kernel)**.** *The Markov kernel $T : \mathcal{O} \to \tilde{\mathcal{O}}$ is reconstructible if there exists a linear mapping $Q : (\mathbb{R}^{\tilde{\mathcal{O}}})^* \to (\mathbb{R}^{\mathcal{O}})^*$, such that $QT = 1$, where Q is known as the* reconstruction.

An immediate consequence of the reconstructible prop-

• *M. Jin, R. Jia, and C. J. Spanos are with the Department of Electrical Engineering and Computer Science, University of California, Berkeley, CA, 94720, United States*
E-mails: {jinming, ruoxijia, spanos}@eecs.berkeley.edu

Manuscript received ; revised

	Static	Chg/Th	Mag/Th	Prc/Th	NaïveBayes	Logistics	AdaBoost	J48	RandForest	SVM
u8	83 (76/98)	72 (87/36)	86 (97/62)	69 (79/45)	95 (96/93)	97 (98/94)	96 (99/88)	**99** (99/98)	**99** (99/98)	**98** (99/96)
u17	65 (65/66)	72 (69/36)	86 (92/76)	71 (83/49)	91 (89/95)	94 (96/90)	93 (98/82)	**98** (99/98)	**99** (99/98)	**98** (98/98)
u20	82 (77/93)	67 (67/67)	86 (94/69)	72 (87/40)	90 (89/90)	91 (93/86)	86 (87/85)	**98** (98/98)	**99** (99/98)	96 (95/97)
u26	81 (74/99)	62 (80/14)	90 (99/66)	67 (87/15)	92 (96/80)	92 (95/84)	91 (96/76)	**99** (99/98)	**99** (99/98)	**96** (95/96)

TABLE 1: BL results for PC dataset, obtained by 10-fold cross-validation, where users (u8, u17, u20, u26) are anonymized. The reporting format follows: overall accuracy (TNR / TPR). The methods, except for Chg/Th, Mag/Th, Prc/Th, are implemented by Weka Machine Learning Toolkit [1]. The static schedule is implemented by assuming occupancy between 8am to 6pm and vacancy otherwise.

	Static	Chg/Th	Mag/Th	Prc/Th	NaïveBayes	Logistic	AdaBoost	J48	RandForest	SVM
r1	**76** (87/73)	65 (30/73)	37 (92/26)	53 (44/55)	79 (88/77)	85 (59/91)	82 (0/100)	**87** (67/92)	**88** (58/94)	**87** (79/89)
r2	72 (69/74)	61 (56/63)	52 (89/39)	52 (52/51)	51 (98/34)	**92** (87/93)	74 (0/100)	**91** (85/93)	**92** (86/94)	**91** (91/91)
r3	69 (63/70)	48 (42/50)	42 (93/25)	51 (52/50)	49 (91/36)	79 (40/91)	79 (23/96)	**83** (66/88)	**84** (58/91)	**82** (69/86)
r4	66 (72/65)	49 (57/49)	24 (98/18)	51 (57/50)	54 (95/51)	**92** (3/99)	**92** (0/100)	**92** (7/99)	**92** (10/98)	**93** (25/98)
r5	**66** (73/65)	54 (40/55)	44 (27/46)	53 (39/54)	70 (89/68)	**92** (0/100)	**92** (0/100)	**92** (0/100)	**93** (2/99)	85 (26/90)

TABLE 2: BL results for ECO dataset by 10-fold cross-validation, where households are anonymized, similar to Table 1. The static schedule is implemented by assuming vacancy between 8am to 6pm and occupancy otherwise.

	Static	MIT	γ-weighted	Unbiased
u8	83 (76/98)	89 (97/71)	**95** (99/85)	**95** (99/84)
u17	65 (65/66)	87 (92/77)	92 (97/81)	**93** (97/84)
u20	82 (77/93)	**87** (94/68)	86 (88/81)	86 (89/80)
u26	81 (74/99)	**93** (96/84)	89 (97/66)	89 (97/68)
r1	76 (87/73)	74 (5/89)	**83** (52/89)	**83** (33/94)
r2	72 (69/74)	74 (1/94)	75 (52/83)	**77** (39/91)
r3	69 (63/70)	76 (50/82)	**78** (54/85)	77 (28/94)
r4	66 (72/65)	**92** (1/99)	70 (25/84)	**92** (1/99)
r5	66 (73/65)	67 (71/66)	70 (17/86)	**93** (1/99)

TABLE 3: NL results for PC and ECO, showing the accuracy (TNR / TPR) for MIT, γ-weighted, and unbiased losses.

	SVM	Transfer A	Transfer B	Transfer C
r1	18 (99/0)	**82** (26/94)	**83** (24/96)	**82** (26/95)
r2	48 (71/39)	77 (42/89)	78 (21/97)	**80** (41/94)
r3	26 (96/5)	75 (46/91)	**76** (41/95)	**75** (45/92)
r4	10 (97/3)	90 (8/97)	**92** (1/100)	**91** (1/98)
r5	15 (91/9)	91 (6/98)	**93** (1/100)	**92** (1/98)

TABLE 4: TL results when only 1% of target dataset is available, averaged over 100 independent trials.

erty is that we have:

$$\langle \mathbb{D}, l(\cdot, f(\cdot))\rangle = \langle QT(\mathbb{D}), l(\cdot, f(\cdot))\rangle = \langle \mathbb{D}, Q^*(l(\cdot, f(\cdot)))\rangle \quad (2)$$

where $\mathbb{D}$ is the original data distribution, $T(\mathbb{D})$ is the corrupted distribution, $Q^*(l(\cdot, f(\cdot)))$ is the corruption corrected loss function, and $\langle \mathbb{D}, l(\cdot, f(\cdot))\rangle = \mathbb{E}_{(\boldsymbol{x},y)\sim\mathbb{D}}\, l(y, f(\boldsymbol{x}))$ is the expectation under the distribution $\mathbb{D}$. The above property implies that working with the corrupted data with $Q^*(l(\cdot, f(\cdot)))$ is equivalent to using the clean data with the original loss function $l(\cdot, f(\cdot))$ associated with learner $f \in \mathcal{F}$.

Since we are starting with noisy labels estimated by the occupancy schedules, the corruption process is characterized by $\rho_{+1} = P(\tilde{y} = -1|y = +1)$ and $\rho_{-1} = P(\tilde{y} = +1|y = -1)$; therefore, we can specify the Markov kernel T and Q^* as:

$$T = \begin{pmatrix} 1-\rho_{-1} & \rho_{+1} \\ \rho_{-1} & 1-\rho_{+1} \end{pmatrix}, \quad (3)$$

$$Q^* = \frac{1}{1-\rho_{-1}-\rho_{+1}} \begin{pmatrix} 1-\rho_{+1} & -\rho_{-1} \\ -\rho_{+1} & 1-\rho_{-1} \end{pmatrix} \quad (4)$$

where Q^* is the conjugate transpose of Q, and it can be verified as the *reconstruction* of T, i.e., $QT = 1$. With elementary calculations, the surrogate loss (9) in the main text follows.

Appendix C Definitions of Divergence Metrics

The more separable the feature, the easier it is to classify [2]. The following divergence metrics for distributions have been employed to measure feature separability, where we use $\boldsymbol{p}$ and $\boldsymbol{q}$ to denote the discrete probabilities:

- Jensen-Shannon divergence:
$$\Upsilon_{JS}(\boldsymbol{p}, \boldsymbol{q}) = \frac{1}{2}\Upsilon_{KL}(\boldsymbol{p}, \boldsymbol{m}) + \frac{1}{2}\Upsilon_{KL}(\boldsymbol{q}, \boldsymbol{m}) \quad (5)$$
where $\boldsymbol{m} = \frac{1}{2}(\boldsymbol{p} + \boldsymbol{q})$, and $\Upsilon_{KL}(\boldsymbol{p}, \boldsymbol{m}) = \sum_k \boldsymbol{p}(k) \ln \frac{\boldsymbol{p}(k)}{\boldsymbol{m}(k)}$ is the Kullback-Leibler divergence.
- Hellinger distance:
$$\Upsilon_{HD}(\boldsymbol{p}, \boldsymbol{q}) = \frac{1}{\sqrt{2}}\sqrt{\sum_k \left(\sqrt{\boldsymbol{p}(k)} - \sqrt{\boldsymbol{q}(k)}\right)^2} \quad (6)$$
- Total variation distance:
$$\Upsilon_{TV}(\boldsymbol{p}, \boldsymbol{q}) = \frac{1}{2}\sum_k |\boldsymbol{p}(k) - \boldsymbol{q}(k)| \quad (7)$$
- Bhattacharyya distance:
$$\Upsilon_{BD}(\boldsymbol{p}, \boldsymbol{q}) = 1 - \sum_i \sqrt{\boldsymbol{p}(k)\boldsymbol{q}(k)} \quad (8)$$

Appendix D Results for BL, NL, and TL

Additional results are listed for the evaluations of BL (Tables 1, 2), NL (Table 3), and TL (Tables 4) on PC and ECO datasets. For the detailed setup and implementations, please refer to the main text.

APPENDIX E

E.1 Proof of Lemma 3.2

Lemma 3.2. *The training noise rate η_t and hypothesis classification error ϵ_t can be estimated assuming we have access to any two of the following quantities:*

a) *(prior information) the number of negative samples, namely $|L^t_{-1,\checkmark}|+|L^t_{+1,\times}|+|U^t_{-1}|$, or $|L^t_{-1,\checkmark}|+|L^t_{+1,\times}|$ for the labeled set*
b) *(Type I or II error) the misclassification rate for either the positive or negative samples, namely $\frac{|L^t_{-1,\times}|}{|L^t_{-1,\times}|+|L^t_{+1,\checkmark}|}$ or $\frac{|L^t_{+1,\times}|}{|L^t_{+1,\times}|+|L^t_{-1,\checkmark}|}$.*

Proof. (Sketch) According to the update rule:

$$L^{t+1}_j = \{L^t_j \cap L^t_{j,n}\} \cup Sample\{L^t_j \Delta L^t_{j,n}; \alpha_j\}, \tag{9}$$

the number of elements in the labeled set of the next iteration depends on the current iteration as follow:

$$|L^{t+1}_{-1,\checkmark}| = |L^t_{-1,\checkmark}|(1-\epsilon_t) + (|L^t_{+1,\times}|+|U^t_{-1}|)(1-\epsilon_t)\alpha_{-1} \tag{10}$$
$$|L^{t+1}_{-1,\times}| = |L^t_{-1,\times}|\epsilon_t + (|L^t_{+1,\checkmark}| + |U^t_{+1}|)\epsilon_t\alpha_{-1} \tag{11}$$
$$|L^{t+1}_{+1,\checkmark}| = |L^t_{+1,\checkmark}|(1-\epsilon_t) + (|L^t_{-1,\times}|+|U^t_{+1}|)(1-\epsilon_t)\alpha_{+1} \tag{12}$$
$$|L^{t+1}_{+1,\times}| = |L^t_{+1,\times}|\epsilon_t + (|L^t_{-1,\checkmark}| + |U^t_{-1}|)\epsilon_t\alpha_{+1} \tag{13}$$

Since we can observe the number of samples in $|L^t_{-1}| = |L^t_{-1,\checkmark}| + |L^t_{-1,\times}|$, $|L^t_{+1}| = |L^t_{+1,\checkmark}| + |L^t_{+1,\times}|$, and $|U^t| = |U^t_{-1}| + |U^t_{+1}|$, and also for those in round $t+1$, we can sum the pairs of (10, 11), also (12, 13). Together with two of the quantities proposed in Lemma 3.2, we can solve the system of equations for the estimation of ϵ_t and η_t. □

As a general remark, the system of equations to be solved in Lemma 3.2 is non-linear, which makes it computationally costly to solve. Since the problem is defined for $0 \le \epsilon_t \le 1$, we can perform a line search of ϵ_t. Given the value of ϵ_t, the system becomes linear and is very easy to solve by taking the inverse, or constrained quadratic programming. Then the optimal ϵ_t that corresponds to the solution that best fits the remaining single equation should be chosen.

E.2 Proof of Theorem 3.3

Theorem 3.3. *If we draw a sequence of*

$$n \ge \frac{2}{\epsilon^2 (1-2\eta)^2} \log\left(\frac{2N}{\delta}\right) \tag{14}$$

samples from a distribution and find any hypothesis f_i that minimizes disagreement with the training labels, where ϵ denotes the hypothesis worst-case classification error rate, η is the upper bound on the training noise rate, N is the number of hypotheses, and δ is the confidence, then the following PAC property is satisfied:

$$\mathbb{P}\left[d\left(f_i, f^*\right) \ge \epsilon\right] \le \delta \tag{15}$$

where $d(,)$ is the sum over the probability of elements from the symmetric difference between the two hypothesis sets f_i and the ground-truth f^*.

Proof. (Sketch) Let $c = 2\mu \log\left(\frac{2N}{\delta}\right)$ where μ is chosen to make the equality holds in (14), then we have $n_t = \frac{c}{\epsilon_t^2(1-2\eta_t)}$, where $m_t = |L^{t+1}_{-1}| + |L^{t+1}_{+1}|$ is the number of samples in the labeled set. We introduce u_t as follow for the simplicity of computation:

$$u_t = \frac{c}{\epsilon_t^2} = n_t\left(1-2\eta_t\right)^2 \tag{16}$$

Since u_t is proportional to $1/\epsilon_t^2$, we have $\epsilon_{t+1} < \epsilon_t$ given that $u_{t+1} > u_t$, thus the assertion follows. □

E.3 Proof of Lemma 3.8

Lemma 3.8. *It can be shown that $\boldsymbol{\omega}_0^* = \frac{\lambda_1}{K\lambda_2}\sum_{k=1}^K \boldsymbol{v}_k^*$.*

The derivation is based on [3] by minimizing the augamented Lagrangian function:

$$L(\boldsymbol{\omega}_0, \boldsymbol{v}_k, \alpha_i^{(k)}\gamma_i^{(k)}) = J_1(\xi_i^{(k)}) + J_2(\boldsymbol{\omega}_0, \boldsymbol{v}_k) - \sum_{k=1}^K \Bigg[\sum_{i=1}^{m(k)} \gamma_i^{(k)}\xi_i^{(k)} + \sum_{i=1}^{m(k)} \alpha_{ik}\left(y_i^{(k)}\langle \boldsymbol{\omega}_0 + \boldsymbol{v}_k, \boldsymbol{x}_i^{(k)}\rangle - 1 + \xi_i^{(k)}\right)\Bigg]$$

from which we can derive that $\boldsymbol{\omega}_0^* = \frac{1}{2\lambda_2}\sum_{k=1}^K\sum_{i=1}^{m(k)}\alpha_i^{(k)}\boldsymbol{x}_i^{(k)}$, and $\boldsymbol{v}_k^* = \frac{T}{2\lambda_1}\sum_{i=1}^{m(k)}\alpha_i^{(k)}\boldsymbol{x}_i^{(k)}$.

REFERENCES

[1] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The weka data mining software: an update," *ACM SIGKDD explorations newsletter*, vol. 11, no. 1, pp. 10–18, 2009.
[2] I. H. Witten and E. Frank, *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, 2005.
[3] T. Evgeniou and M. Pontil, "Regularized multi–task learning," in *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2004, pp. 109–117.