Unsupervised Ensemble Classification With Sequential and Networked Data

Panagiotis A. Traganitis⬡, *Member, IEEE* and Georgios B. Giannakis⬡, *Fellow, IEEE*

Abstract—Ensemble learning, the machine learning paradigm where multiple models are combined, has exhibited promising perfomance in a variety of tasks. The present work focuses on unsupervised ensemble classification. The term unsupervised refers to the ensemble combiner who has no knowledge of the ground-truth labels that each classifier has been trained on. While most prior works on unsupervised ensemble classification are designed for independent and identically distributed (i.i.d.) data, the present work introduces an unsupervised scheme for learning from ensembles of classifiers in the presence of data dependencies. Two types of data dependencies are considered: sequential data and networked data whose dependencies are captured by a graph. For both, novel moment matching and Expectation-Maximization algorithms are developed. Performance of these algorithms is evaluated on synthetic and real datasets, which indicate that knowledge of data dependencies in the meta-learner is beneficial for the unsupervised ensemble classification task.

Index Terms—Ensemble learning, unsupervised, sequential classification, crowdsourcing, dependent data

✦

1 INTRODUCTION

As social networks, connected "smart" devices and highly accurate scientific instruments have permeated society, multiple machine learning, signal processing and data mining algorithms have been developed to process the generated data and draw inferences from them. With most of these algorithms typically designed to operate under different assumptions, combining them can be beneficial because different algorithms can complement each others strengths.

Ensemble learning refers to the task of combining multiple, possibly heterogeneous, machine learning models or learners [1], [2].[1] In particular, ensemble classification refers to combining different classifiers [3], [4]. Most popular ensemble approaches, such as boosting, bagging and stacking, have been developed for the supervised setup. Boosting methods improve an ensemble of weak learners by iteratively retraining them and properly weighting their outputs [5]. Bagging algorithms train an ensemble of base classifiers on bootstrap samples of a dataset, and combine their results using majority voting. Stacking approaches train a *meta-learner* using the outputs of base learners as features [6], [7].

In many cases however, labeled data may not be available, and/or one may only have access to pre-trained

classifiers or human annotators, justifying the need for *unsupervised* ensemble learning methods. Such a setup emerges in diverse disciplines including medicine [8], biology [9], team decision making, distributed detection, and economics [10], and has recently gained attention with the advent of crowdsourcing [11], as well as services such as Amazon's Mechanical Turk [12], and Figure 8 [13], to name a few. Particularly in crowdsourcing, human annotators play the role of base classifiers of an ensemble learning model. Unsupervised ensemble classification is similar to stacking: a *meta-learner* has to learn how to combine the outputs of multiple base learners, in the absence of labeled data.

Multiple algorithms attempt to address the unsupervised ensemble classification problem, and a common assumption is that the data are independent and identically distributed (i.i.d.) from an unknown distribution [14], [15], [16], [17], [18], [19], [20]. In several cases however, additional domain knowledge may be available to the meta-learner. This domain knowledge provides information regarding the data distribution, as well as data dependencies. In this paper, two types of data dependence are considered: sequential and networked data. Classification of sequential data arises in many natural language processing tasks such as part-of-speech tagging, named-entity recognition, and information extraction, to name a few [21]. Examples of networked data, where data correlations or dependencies are captured in a known graph, include citation, social, communication and brain networks among others. If data do not exhibit network structure, the proposed networked data models can accommodate side information in the form of a graph to enhance the label fusion process.

A novel unified framework for *unsupervised ensemble classification with data dependencies* is proposed. The presented methods and algorithms are built upon simple concepts from probability, as well as recent advances in tensor decompositions [22] and optimization theory, that enable

---

1. The terms learner, model, and classifier will be used interchangeably throughout this manuscript.

• *The authors are with the Department of Electrical and Computer Engineering and the Digital Technology Center, University of Minnesota, Minneapolis, MN 55455, USA. E-mail: {traga003, georgios}@umn.edu.*

assessing the reliability of multiple learners, and combining their answers. Similar to prior works for i.i.d. data, in our proposed model each learner has a fixed probability of deciding that a datum belongs to class k, given that the true class of the datum is k'. These probabilities parametrize the learners. Data dependencies are then encoded in the marginal probability mass function (pmf) of the data labels. For sequential data, the pmf of data labels is assumed to be a Markov chain, while for networked data the pmf of data labels is assumed to be a Markov Random Field (MRF.) As we are operating in an unsupervised regime, it is required to assume that learners make decisions independent of each other. This assumption provides analytical and computational tractability. Under this assumption, the proposed methods are able to extract the probabilities that parametrize learner performance from their responses. As an initial step of the proposed framework, the moment-matching method we introduced in [18] for ensemble classification of i.i.d. data, is adopted to provide rough estimates for learner parameters. At the second step, our newly developed expectation maximization (EM) algorithms are employed to refine the parameters obtained from the initial moment matching step. These EM algorithms are tailored for the dependencies present in the data and produce the final label estimates.

The rest of the paper is organized as follows. Section 2 states the problem, and provides preliminaries along with a brief description of the prior art in unsupervised ensemble classification for i.i.d. data. Section 3 provides an outline of moment matching and EM methods for i.i.d. data; Section 4 introduces the proposed approach to unsupervised ensemble classification for sequential data; while Section 5 deals with its counterpart for networked data. Section 6 presents numerical tests to evaluate our methods. Finally, concluding remarks and future research directions are given in Section 7.

Notation. Unless otherwise noted, lowercase bold letters, x, denote vectors, uppercase bold letters, $\mathbf{X}$, represent matrices, and calligraphic uppercase letters, $\mathcal{X}$, stand for sets. The (i,j)th entry of matrix $\mathbf{X}$ is denoted by $[\mathbf{X}]_{ij}$; $\mathbf{X}^\top$ denotes the tranpose of matrix $\mathbf{X}$; $\mathbb{R}^D$ stands for the D-dimensional real euclidean space, $\mathbb{R}_+$ for the set of positive real numbers, $\mathbb{E}[\cdot]$ for expectation, and $\|\cdot\|$ for the ℓ_2-norm. Underlined capital letters $\underline{X}$ denote tensors; while $[[\mathbf{A},\mathbf{B},\mathbf{C}]]_K$ is used to denote compactly a K parallel factor (PARAFAC) analysis tensor [22] with factor matrices $\mathbf{A} := [a_1,\ldots,a_K]$, $\mathbf{B} := [b_1,\ldots,b_K]$, $\mathbf{C} := [c_1,\ldots,c_K]$, that is $[[\mathbf{A},\mathbf{B},\mathbf{C}]]_K = \sum_{k=1}^K a_k \circ b_k \circ c_k$. Finally, $\mathcal{I}(A)$ denotes the indicator function of event A, i.e., $\mathcal{I}(A) = 1$ if A occurs and is 0 otherwise.

2 PROBLEM STATEMENT AND PRELIMINARIES

Consider a dataset consisting of N data (possibly vectors) $\{x_n\}_{n=1}^N$ each belonging to one of K possible classes with corresponding labels $\{y_n\}_{n=1}^N$, e.g., $y_n = k$ if x_n belongs to class k. The pairs $\{(x_n, y_n)\}_{n=1}^N$ are drawn from an unknown joint distribution D, and X and Y denote random variables such that $(X, Y) \sim$ D. Consider now M pre-trained learners that observe $\{x_n\}_{n=1}^N$, and provide estimates of labels. Let $f_m(x_n) \in \{1,\ldots,K\}$ denote the label assigned to datum x_n by the mth learner. All learner responses are then collected at a meta-learner. Collect all learner responses in the $M \times N$
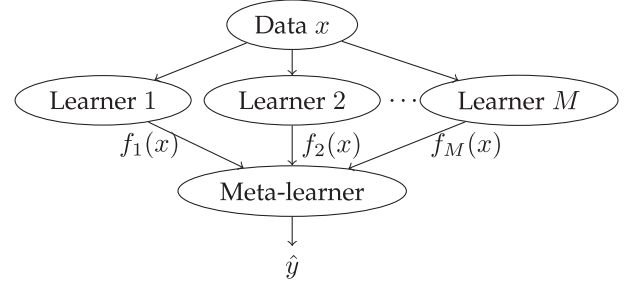

Fig. 1. Unsupervised ensemble classification setup, where the outputs of learners are combined in parallel.

matrix $\mathbf{F}$, that has entries $[\mathbf{F}]_{mn} = f_m(x_n)$, and all ground-truth labels in the $N \times 1$ vector $y = [y_1,\ldots,y_N]^\top$. The task of *unsupervised ensemble classification* is: Given only the learner responses $\{f_m(x_n), m = 1,\ldots,M\}_{n=1}^N$, we wish to estimate the ground-truth labels of the data $\{y_n\}$; see Fig. 1. In contrast to supervised ensemble classification, here the combined models (learners) have been trained beforehand. In addition, the meta-learner does not have access to any ground-truth data to learn a combining function. Thus, the focus of this work is on judiciously combining learner responses. This setup is similar to crowdsourced classification, which also has the additional challenge that human annotators (corresponding to the learners in the unsupervised ensemble model) may not provide responses for all N data.

Consider that each learner has a fixed probability of deciding that a datum belongs to class k', when presented with a datum of class k; and all classifiers behavior is presumed invariant throughout the dataset. Clearly, the performance of each learner m is then characterized by the so-called *confusion* matrix $\mathbf{\Gamma}_m$, whose (k', k)th entry is

$$[\mathbf{\Gamma}_m]_{k'k} := \Gamma_m(k', k) = \Pr(f_m(X) = k'|Y = k). \quad (1)$$

The $K \times K$ matrix $\mathbf{\Gamma}_m$ has non-negative entries that obey the simplex constraint, since $\sum_{k'=1}^K \Pr(f_m(X) = k'|Y = k) = 1$, for $k = 1,\ldots,K$; hence, entries of each $\mathbf{\Gamma}_m$ column sum up to 1, that is, $\mathbf{\Gamma}_m^\top 1 = 1$ and $\mathbf{\Gamma}_m \geq 0$. The confusion matrix showcases the statistical behavior of a learner, as each column provides the learners's probability of deciding the correct class, when presented with a datum from each class.

Before proceeding, we adopt the following assumptions.

As1 Responses of different learners per datum are conditionally independent, given the ground-truth label Y of the same datum X; that is

$$\Pr(f_1(X) = k_1,\ldots,f_M(X) = k_M|Y = k)$$
$$= \prod_{m=1}^M \Pr(f_m(X) = k_m|Y = k) = \prod_{m=1}^M \Gamma_m(k_m, k).$$

As2 Most learners are better than random.

As1 is satisfied by learners making decisions independently, which is a rather standard assumption [14], [16], [17]. As1 can be thought of as the manifestation of diversity, which is sought after in ensemble learning systems [2]. Graphically, this model is depicted in Fig. 2. Under As2, the largest entry per $\mathbf{\Gamma}_m$ column is the one on the diagonal
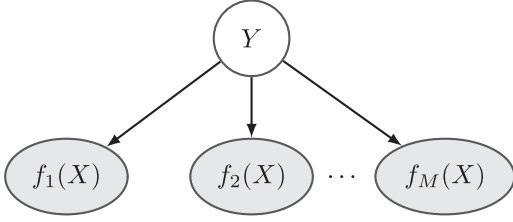
Fig. 2. Graphical depiction of the Dawid-Skene model for i.i.d. data. Shaded ellipses are observed variables (here classifier responses).

$$[\mathbf{\Gamma}_m]_{kk} \geq [\mathbf{\Gamma}_m]_{k'k}, \text{ for } k', k = 1, \ldots, K.$$

As2 is another standard assumption, used to alleviate the inherent permutation ambiguity in estimating $\mathbf{\Gamma}_m$.

Based on the aforementioned model, knowledge of the structure of the joint pdf of learner responses and labels is critical. When data $\{(x_n, y_n)\}_{n=1}^{N}$ are i.i.d. the joint pdf can be expressed as a product, that is

$$\Pr(\mathbf{F}, \boldsymbol{y}) = \prod_{n=1}^{N} \Pr(\mathbf{f}_n, y_n), \tag{2}$$

where $\mathbf{f}_n := [f_1(x_n), \ldots, f_M(x_n)]^{\top}$.

Consider now that data $\{(x_n, y_n)\}_{n=1}^{N}$ belong to a sequence, i.e., the nth datum depends on the $(n-1)$st one. Pertinent settings emerge with speech and natural language processing tasks such as word identification, part-of-speech tagging, named-entity recognition, and information extraction [21].

In order to take advantage of this dependence, we will encode it in the marginal pmf $\Pr(\boldsymbol{y})$ of the labels. Specifically, we postulate that the sequence of labels $\{y_n\}_{n=1}^{N}$ forms a one-step time-homogeneous Markov chain; that is, variable y_n depends only on its immediate predecessor y_{n-1}. This Markov chain is characterized by a $K \times K$ transition matrix $\mathbf{T}$, whose (k, k')th entry is given by

$$[\mathbf{T}]_{kk'} = T(k, k') = \Pr(y_n = k | y_{n-1} = k').$$

Matrix $\mathbf{T}$ has non-negative entries that satisfy the simplex constraint. The marginal probability of $\{y_n\}_{n=1}^{N}$ can be expressed using successive conditioning, as

$$\Pr(\boldsymbol{y} = \boldsymbol{k}) = \Pr(y_1 = k_1) \prod_{n=2}^{N} T(k_n, k_{n-1}), \tag{3}$$

where $\boldsymbol{k} := [k_1, \ldots, k_N]^{\top}$. Accordingly, the data $\{x_n\}_{n=1}^{N}$ depend only on their corresponding y_n, and can be drawn from an unknown conditional pdf as $x_n \sim \Pr(x_n | y_n = k_n)$. The data pairs $\{(x_n, y_n)\}_{n=1}^{N}$ form a hidden Markov model (HMM), where the labels $\{y_n\}_{n=1}^{N}$ correspond to the hidden variables of the HMM, while $\{x_n\}_{n=1}^{N}$ correspond to the

observed variables of the HMM. As with the i.i.d. case, M learners observe $\{x_n\}_{n=1}^{N}$, and provide estimates of their labels $f_m(x_n)$. Under As1, the responses of different learners per datum are conditionally independent, given the ground-truth label y_n of the same datum x_n. A graphical representation of this HMM is provided in Fig. 3.

Another type of data dependency considered in this work, is the dependency between networked data. In many cases, additional information pertaining to the data is available in the form of an undirected graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ and $\mathcal{E}$ denote the vertex (or node) and edge sets of $\mathcal{G}$, respectively. Examples of such networked data include social and citation networks [23], [24]. Each node of this graph corresponds to a data point, thus $|\mathcal{V}| = N$, while the edges capture pairwise relationships between the data.

For networked data, we will take a similar approach to sequential data and encode data dependence, meaning the pairwise relationships provided by the graph $\mathcal{G}$, in the marginal pmf $\Pr(\boldsymbol{y})$. Specifically, we model the labels $\{y_n\}_{n=1}^{N}$ as being drawn from an MRF. Due to the MRF structure of $\Pr(\boldsymbol{y})$, the conditional pmf of y_n, for all $n = 1, \ldots, N$, satisfies the local Markov property

$$\Pr(y_n | \boldsymbol{y}_{-n}) = \Pr(y_n | \boldsymbol{y}_{\mathcal{N}_n}), \tag{4}$$

where $\boldsymbol{y}_{-n}$ is a vector containing all labels except y_n and $\boldsymbol{y}_{\mathcal{N}_n}$ is a vector containing the labels of node n neighbors. Then, the joint pmf of all labels is

$$\Pr(\boldsymbol{y}) = \frac{1}{Z} \exp(-U(\boldsymbol{y})), \tag{5}$$

where $Z := \sum_{\boldsymbol{y}} \exp(-U(\boldsymbol{y}))$ is the normalization constant, and $U(\boldsymbol{y})$ is the so-called energy function. Computing the normalization constant Z involves all possible configurations of $\boldsymbol{y}$; hence, it is intractable for datasets with moderate to large size N. By the Hammersley-Clifford theorem the energy function can be written as [25]

$$U(\boldsymbol{y}) = \frac{1}{2} \sum_{(n,n') \in \mathcal{E}} V(y_n, y_{n'}), \tag{6}$$

where $V(y_n, y_{n'})$ denotes the clique potential of the (n, n')th edge, which will be defined in the following sections.

Similar to the i.i.d. and sequential cases, data vectors $\{x_n\}_{n=1}^{N}$ are generated from an unknown conditional pdf $x_n \sim \Pr(x_n | y_n = k)$ that depends only on their corresponding label y_n. M learners observe $\{x_n\}_{n=1}^{N}$ and provide estimates of their labels $f_m(x_n)$. With As1 still in effect, learner responses are conditionally independent given Y.
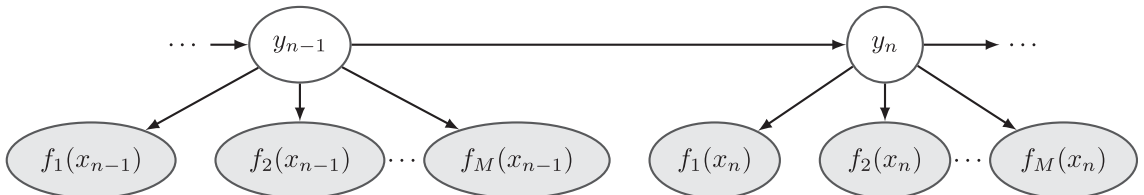

Fig. 3. Graphical representation of the proposed model for sequential data. Shaded ellipses indicate observed variables (here learner responses).

2.1 Prior Work

Most prior works on unsupervised ensemble classification focus on the i.i.d. data case. Possibly the simplest scheme is majority voting, where the estimated label of a datum is the one that most learners agree upon. Majority voting presumes that all learners are equally "reliable," which is rather unrealistic, both in crowdsourcing as well as in ensemble learning setups. Other unsupervised ensemble methods aim to estimate the parameters that characterize the learners' performance, namely the learner confusion matrices. A popular approach is joint maximum likelihood (ML) estimation of the unknown labels and the aforementioned confusion matrices using the expectation-maximization (EM) algorithm [14]. As EM iterations do not guarantee convergence to the ML solution, recent works pursue alternative estimation methods. For binary classification, [15], [26] describe each learner with one parameter, its probability of providing a correct response, and attempts to learn this parameter. Recently, for binary classification, [17] used the eigendecomposition of the learner crosscovariance matrix, to estimate the entries of the 2×2 confusion matrices of learners, while [27] introduced a minimax optimal algorithm that can infer learner reliabilities.. In the multiclass setting, spectral methods such as [16], [18] utilize third-order moments and tensor decompositions to estimate the unknown reliability parameters. These estimates can then initialize the EM algorithm of [14].

Recent works advocate unsupervised ensemble approaches for sequential data. A method to aggregate learner labels for sequential data relies on conditional random fields (CRFs) [28]. However, this method operates under strong and possibly less realistic assumptions requiring e.g., that only one learner provides the correct label for each datum. To relax the assumptions of [28], extensions of the standard Hidden Markov Model (HMM) to incorporate learner responses have been reported along with a variational EM [29] algorithm to aggregate them [30]. As both aforementioned methods require tuning of hyperparameters, a training step is necessary, which can be unrealistic in unsupervised settings.

When features $\{x_n\}_{n=1}^N$ are available at the meta-learner, approaches based on Gaussian Processes can be used to classify the data based on learner responses [31], [32], [33]. These approaches can take advantage of some data dependencies, as they rely on linear or nonlinear similarities between data; however, in addition to requiring the data features at the meta-learner, these methods have only been developed for binary classification.

The present work puts forth a novel scheme for *unsupervised ensemble classification in the presence of data dependencies*. Our approach builds upon our previous work on unsupervised ensemble classification of i.i.d. data [18], [19], and markedly extends its scope to handle sequential as well as networked data, without requiring training or access to data features. For sequentially dependent data case we present a moment matching algorithm that is able to estimate learner confusion matrices as well as the parameters characterizing the Markov chain of the labels. These confusion matrices and parameters are then refined using EM iterations tailored for the sequential data classification task. For the network dependencies, the moment matching method for i.i.d. data is used to initialize an EM algorithm designed

for networked data. To our knowledge, no existing work tackles the ensemble classification task under the networked data regime. Compared to our conference precursor in [34], here we have included extensive numerical tests as well as a new EM algorithm for sequential data along with algorithms that tackle networked data.

3 UNSUPERVISED ENSEMBLE CLASSIFICATION OF I.I.D. DATA

Before introducing algorithms for sequential and networked data, we first outline moment matching and EM approaches for i.i.d. data. If all learner confusion matrices were ideally known, the label of x_n could be estimated using a maximum a posteriori (MAP) classifier. The latter finds the label k that maximizes the joint probability of y_n and observed learner responses $\{f_m(x_n) = k_m\}_{m=1}^M$

$$
\begin{aligned}
\hat{y}_n &= \underset{k \in \{1,\dots,K\}}{\arg \max} \, L(x_n | k) \Pr(y_n = k) \\
&= \underset{k \in \{1,\dots,K\}}{\arg \max} \, \log \pi_k + \sum_{m=1}^M \log \left(\Gamma_m(k_m, k) \right),
\end{aligned}
\tag{7}
$$

where $\pi_k := \Pr(y_n = k) = \Pr(Y = k)$ and $L(x_n | k) := \Pr(f_1(x_n) = k_1, \dots, f_M(x_n) = k_M | Y = k)$, and the second equality of (7) follows from As1 and properties of the logarithm. In addition, if all classes are assumed equiprobable, (7) reduces to the maximum likelihood (ML) classifier.

But even for non-equiprobable classes, unsupervised ensemble classification requires estimates of the class priors $\boldsymbol{\pi} := [\pi_1, \dots, \pi_K]^\top$ as well as all the learner performance parameters $\{\boldsymbol{\Gamma}_m\}_{m=1}^M$.

3.1 EM Algorithm for i.i.d. Data

Here we outline how the EM algorithm can be employed to estimate the wanted learner performance parameters by iteratively maximizing the log-likelihood of the observed learner responses; that is, $\log \Pr(\mathbf{F}; \boldsymbol{\theta})$, where $\boldsymbol{\theta}$ collects all the learner confusion matrices (prior probabilities are assumed equal and are dropped for simplicity). Each EM iteration includes the expectation (E-)step and the maximization (M-)step.

At the E-step of the $(i+1)$st iteration, the estimate $\boldsymbol{\theta}^{(i)}$ is given, and the so-termed Q-function is obtained as

$$
\begin{aligned}
Q(\boldsymbol{\theta}; \boldsymbol{\theta}^{(i)}) &= \mathbb{E}_{\boldsymbol{y}|\mathbf{F}; \boldsymbol{\theta}^{(i)}}[\log \Pr(\boldsymbol{y}, \mathbf{F}; \boldsymbol{\theta})] \\
&= \mathbb{E}_{\boldsymbol{y}|\mathbf{F}; \boldsymbol{\theta}^{(i)}}[\log \Pr(\mathbf{F}|\boldsymbol{y}; \boldsymbol{\theta})] + \mathbb{E}_{\boldsymbol{y}|\mathbf{F}; \boldsymbol{\theta}^{(i)}}[\log \Pr(\boldsymbol{y}; \boldsymbol{\theta})].
\end{aligned}
\tag{8}
$$

Since the data are i.i.d., we have under As1 that

$$
\mathbb{E}_{\boldsymbol{y}|\mathbf{F}; \boldsymbol{\theta}^{(i)}}[\log \Pr(\mathbf{F}|\boldsymbol{y}; \boldsymbol{\theta})] = \sum_{n=1}^N \sum_{k=1}^K \sum_{m=1}^M \log \Gamma_m(f_m(x_n), k) q_{nk},
$$

where $q_{nk} := \Pr(y_n = k | \{f_m(x_n)\}_{m=1}^M; \boldsymbol{\theta}^{(i)})$ is the posterior of label y_n given the observed data and current parameters, and

$$
\mathbb{E}_{\boldsymbol{y}|\mathbf{F}; \boldsymbol{\theta}^{(i)}}[\log \Pr(\boldsymbol{y}; \boldsymbol{\theta})] = \sum_{n=1}^N \sum_{k=1}^K \log \Pr(y_n = k; \boldsymbol{\theta}) q_{nk}.
$$

Under this model, it can be shown that [14], [16]

$$q_{nk}^{(i+1)} = \frac{1}{Z} \exp \left(\sum_{m=1}^{M} \sum_{k'=1}^{K} \mathcal{I}(f_m(x_n) = k') \log \left(\Gamma_m^{(i)}(k', k) \right) \right),$$
(9)

where Z is the normalization constant.

At the M-step, learner confusion matrices are updated by maximizing the Q-function to obtain

$$\boldsymbol{\theta}^{(i+1)} = \arg \max_{\boldsymbol{\theta}} Q(\boldsymbol{\theta}; \boldsymbol{\theta}^{(i)}).$$
(10)

It can be shown per learner m that (10) boils down to

$$[\boldsymbol{\Gamma}_m^{(i+1)}]_{k'k} = \frac{\sum_{n=1}^{N} q_{nk}^{(i+1)} \mathcal{I}(f_m(x_n) = k')}{\sum_{k''=1}^{K} \sum_{n=1}^{N} q_{nk}^{(i+1)} \mathcal{I}(f_m(x_n) = k'')}.$$
(11)

The E- and M-steps are repeated until convergence, and ML estimates of the labels are subsequently found as

$$\begin{aligned} \hat{y}(x_n) &= \arg \max_{k \in \{1,\dots,K\}} \Pr(\{f_m(x_n)\}_{m=1}^{M}, y_n = k) \\ &= \arg \max_{k \in \{1,\dots,K\}} q_{nk}. \end{aligned}$$

3.2 Moment-Matching for i.i.d. Data

As an alternative to EM, the annonator performance parameters can be estimated using the moment-matching method we introduced for i.i.d. data in [18], which we outline here before extending it to dependent data in the ensuing sections.

Consider representing label k using the canonical $K \times 1$ vector e_k, namely the kth column of the $K \times K$ identity matrix $\mathbf{I}$. Let $\mathbf{f}_m(X)$ denote the mth learner's response in vector format. Since $\mathbf{f}_m(X)$ is just a vector representation of $f_m(X)$, it holds that $\Pr(f_m(X) = k'|Y = k) \equiv \Pr(\mathbf{f}_m(X) = e_{k'}|Y = k)$. With $\boldsymbol{\gamma}_{m,k}$ denoting the kth column of $\boldsymbol{\Gamma}_m$, it thus holds that

$$\boldsymbol{\gamma}_{m,k} := \mathbb{E}[\mathbf{f}_m(X)|Y = k] = \sum_{k'=1}^{K} e_{k'} \Pr(f_m(X) = k'|Y = k),$$
(12)

where we used the definition of conditional expectation. Using (12) and the law of total probability, the mean vector of responses from learner m, is hence

$$\mathbb{E}[\mathbf{f}_m(X)] = \sum_{k=1}^{K} \mathbb{E}[\mathbf{f}_m(X)|Y = k] \Pr(Y = k) = \boldsymbol{\Gamma}_m \boldsymbol{\pi}.$$
(13)

Upon defining the diagonal matrix $\boldsymbol{\Pi} := \mathrm{diag}(\boldsymbol{\pi})$, the $K \times K$ cross-correlation matrix between the responses of learners m and $m' \neq m$, can be expressed as

$$\begin{aligned} \mathbf{R}_{mm'} &:= \mathbb{E}[\mathbf{f}_m(X)\mathbf{f}_{m'}^{\top}(X)] \\ &= \sum_{k=1}^{K} \mathbb{E}[\mathbf{f}_m(X)|Y = k] \mathbb{E}[\mathbf{f}_{m'}^{\top}(X)|Y = k] \Pr(Y = k) \\ &= \boldsymbol{\Gamma}_m \mathrm{diag}(\boldsymbol{\pi}) \boldsymbol{\Gamma}_{m'}^{\top} = \boldsymbol{\Gamma}_m \boldsymbol{\Pi} \boldsymbol{\Gamma}_{m'}^{\top} \end{aligned}$$
(14)

where we successively relied on the law of total probability, As1, and (12). Consider now the $K \times K \times K$ cross-correlation tensor between the responses of learners m, $m' \neq m$ and $m'' \neq m', m$, namely

$$\underline{\boldsymbol{\Psi}}_{mm'm''} = \mathbb{E}[\mathbf{f}_m(X) \circ \mathbf{f}_{m'}(X) \circ \mathbf{f}_{m''}(X)].$$
(15)

It can be shown that $\underline{\boldsymbol{\Psi}}_{mm'm''}$ obeys a PARAFAC model with factor matrices $\boldsymbol{\Gamma}_m, \boldsymbol{\Gamma}_{m'}$ and $\boldsymbol{\Gamma}_{m''}$ [18]; that is

$$\begin{aligned} \underline{\boldsymbol{\Psi}}_{mm'm''} &= \sum_{k=1}^{K} \pi_k \boldsymbol{\gamma}_{m,k} \circ \boldsymbol{\gamma}_{m',k} \circ \boldsymbol{\gamma}_{m'',k} \\ &= [[\boldsymbol{\Gamma}_m \boldsymbol{\Pi}, \boldsymbol{\Gamma}_{m'}, \boldsymbol{\Gamma}_{m''}]]_K. \end{aligned}$$
(16)

Note here that the diagonal matrix $\boldsymbol{\Pi}$ can multiply any of the factor matrices $\boldsymbol{\Gamma}_m, \boldsymbol{\Gamma}_{m'}$, or, $\boldsymbol{\Gamma}_{m''}$.

Having available the sample average counterparts of (13), (14) and (15), correspondingly $\{\boldsymbol{\mu}_m\}_{m=1}^{M}$, $\{\mathbf{S}_{mm'}\}_{m,m'=1}^{M}$, and $\{\underline{T}_{mm'm''}\}_{m,m',m''=1}^{M}$, estimates of $\{\boldsymbol{\Gamma}_m\}_{m=1}^{M}$ and $\boldsymbol{\pi}$ can be readily obtained. This approach is an instantiation of the method of moments estimation method, see e.g., [35], and can be cast as the following constrained optimization task

$$\min_{\boldsymbol{\pi} \in \mathcal{C}_\pi, \{\boldsymbol{\Gamma}_m \in \mathcal{C}\}_{m=1}^{M}} g(\{\boldsymbol{\Gamma}_m\}_{m=1}^{M}, \boldsymbol{\pi}),$$
(17)

where

$$\begin{aligned} g(\{\boldsymbol{\Gamma}_m\}, \boldsymbol{\pi}) :=& \sum_{m=1}^{M} \|\boldsymbol{\mu}_m - \boldsymbol{\Gamma}_m \boldsymbol{\pi}\|_2^2 \\ &+ \sum_{\substack{m=1 \\ m' > m}}^{M} \|\mathbf{S}_{mm'} - \boldsymbol{\Gamma}_m \boldsymbol{\Pi} \boldsymbol{\Gamma}_{m'}^{\top}\|_F^2 \\ &+ \sum_{\substack{m=1 \\ m' > m, m'' > m'}}^{M} \|\underline{T}_{mm'm''} - [[\boldsymbol{\Gamma}_m \boldsymbol{\Pi}, \boldsymbol{\Gamma}_{m'}, \boldsymbol{\Gamma}_{m''}]]_K\|_F^2, \end{aligned}$$

$\mathcal{C} := \{\boldsymbol{\Gamma} \in \mathbb{R}^{K \times K} : \boldsymbol{\Gamma} \geq 0, \boldsymbol{\Gamma}^{\top} \mathbf{1} = 1\}$, is the convex set of matrices whose columns lie on a probability simplex, and $\mathcal{C}_p := \{\mathbf{u} \in \mathbb{R}^K : \mathbf{u} \geq 0, \mathbf{u}^{\top} \mathbf{1} = 1\}$ denotes the simplex constraint for a $K \times 1$ vector. The non-convex optimization in (17) can be solved using the alternating optimization method described in [18], which is guaranteed to converge to a stationary point of g [36]. As2 is used here to address the permutation ambiguity that is induced by the tensor decomposition of (17). Interested readers are referred to [18] for implementation details, and theoretical guarantees.

Upon obtaining $\{\hat{\boldsymbol{\Gamma}}_m\}_{m=1}^{M}$ and $\hat{\boldsymbol{\pi}}$, a MAP classifier can be subsequently employed to estimate the label for each datum; that is, for $n = 1, \dots, N$, we obtain

$$\hat{y}(x_n) = \arg \max_{k \in \{1,\dots,K\}} \log \hat{\pi}_k + \sum_{m=1}^{M} \log \hat{\Gamma}_m(f_m(x_n), k),$$
(18)

where $\hat{\Gamma}_m(k', k) = [\hat{\boldsymbol{\Gamma}}_m]_{k'k}$, and $\hat{\pi}_k = [\hat{\boldsymbol{\pi}}]_k$. The estimates $\{\hat{\boldsymbol{\Gamma}}_m\}_{m=1}^{M}, \hat{\boldsymbol{\pi}}, \{\hat{y}_n\}$, can be improved using the EM iterations in Section 3.1. Such a refinement is desirable when N is relatively small, and thus moment estimates are not as reliable.

Next, we will introduce our novel approaches for ensemble classification with sequential and networked data.

4 SEQUENTIALLY DEPENDENT DATA

Having recapped moment matching and EM approaches for i.i.d. data, we now turn our attention to the sequential data case. Recall from Section 2 that we postulate labels y forming a one-step time-homogeneous Markov chain, characterized by the transition matrix $\mathbf{T} \in \mathcal{C}$, and that learner responses obey As1. The labels y along with learner responses $\mathbf{F}$, form an HMM. As with the i.i.d. case, here we develop both moment-matching and EM algorithms tailored for sequential data.

4.1 Label Estimation for Sequential Data

Given only learner responses for all data in a sequence, an approach to estimating the labels of each datum, meaning the hidden variables of the HMM, is to find the sequence k that maximizes the joint probability of the labels y and the learner responses $\mathbf{F}$, namely

$$\begin{aligned} &\Pr(\boldsymbol{y} = \boldsymbol{k}, \mathbf{F}) \\ &= \Pr(y_1 = k_1) \prod_{n=2}^{N} T(k_n, k_{n-1}) \prod_{m=1}^{M} \Gamma_m(f_m(x_n), k_n), \end{aligned} \quad (19)$$

where the equality is due to (3) and As1. This can be done efficiently using the Viterbi algorithm [37], [38]. In order to obtain estimates of the labels, $\{\Gamma_m\}_{m=1}^{M}$ and $\mathbf{T}$ must be available. The next subsections will show that $\{\Gamma_m\}_{m=1}^{M}$ and $\mathbf{T}$ can be recovered by the learner responses, using moment matching and/or EM approaches.

4.2 Moment Matching for Sequential Data

Under the sequential data model outlined earlier, we require an additional assumption before presenting our moment matching algorithm.

As3 The Markov chain formed by the labels $\{y_n\}$ has a unique stationary distribution $\boldsymbol{\pi} := [\pi_1, \dots, \pi_K]^\top = [\Pr(Y = 1), \dots, \Pr(Y = K)]^\top$, and is also irreducible.

Similar to [39], this assumption enables decoupling the problem of learning the parameters of interest in two steps. First, estimates of the confusion matrices $\{\hat{\Gamma}_m\}_{m=1}^{M}$ and stationary priors $\hat{\boldsymbol{\pi}}$ are obtained; and subsequently, the transition matrix is estimated as $\hat{\mathbf{T}}$ before obtaining an estimate of the labels $\{\hat{y}_n\}_{n=1}^{N}$.

Under As3, the HMM is mixing and assuming that y_0 is drawn from the stationary distribution $\boldsymbol{\pi}$, the responses of a learner m can be considered to be generated from a mixture model, see e.g., [39]

$$f_m(X) \sim \sum_{k=1}^{K} \pi_k \Pr(f_m(X)|Y = k) . \quad (20)$$

Based on the latter, the remainder of this subsection will treat labels $\{y_n\}_{n=1}^{N}$, as if they had been drawn i.i.d. from the stationary distribution $\boldsymbol{\pi}$, that is $y_n \sim \boldsymbol{\pi}$ for $n = 1, \dots, N$. Then, the procedure outlined in Section 3.2 can be readily adopted to obtain estimates of the stationary distribution $\hat{\boldsymbol{\pi}}$ and the confusion matrices $\{\hat{\Gamma}_m\}_{m=1}^{M}$.

With estimates of learner confusion matrices $\{\hat{\Gamma}_m\}$ and stationary probabilities $\hat{\boldsymbol{\pi}}$ at hand, we turn our attention to the estimation of the transition matrix $\mathbf{T}$. To this end,

consider the cross-correlation matrix of consecutive vectorized observations between learners m and m', namely $\tilde{\mathbf{R}}_{mm'} = \mathbb{E}[\mathbf{f}_m(x_n)\mathbf{f}_{m'}^\top(x_{n-1})]$. Under the HMM of Section 4, the latter is given by

$$\tilde{\mathbf{R}}_{mm'} = \Gamma_m \mathbf{T} \mathrm{diag}(\boldsymbol{\pi}) \Gamma_{m'}^\top = \Gamma_m \mathbf{A} \Gamma_{m'}^\top \quad (21)$$

where $\mathbf{A} := \mathbf{T}\mathrm{diag}(\boldsymbol{\pi})$. Letting $\tilde{\mathbf{S}}_{mm'}$ denote the sample counterpart of (21), and with $\{\hat{\Gamma}_m\}_{m=1}^{M}$ available, we can recover $\mathbf{T}$ as follows. First, we solve the convex moment-matching optimization problem

$$\min_{\mathbf{A} \in \mathcal{C}_S} \sum_{\substack{m=1 \\ m'>m}}^{M} \|\tilde{\mathbf{S}}_{mm'} - \hat{\Gamma}_m \mathbf{A} \hat{\Gamma}_{m'}^\top\|_F^2, \quad (22)$$

where $\mathcal{C}_S$ is the set of matrices whose entries are positive and sum to 1, namely $\mathcal{C}_S := \{\mathbf{X} \in \mathbb{R}^{K \times K} : \mathbf{X} \geq 0, \mathbf{1}^\top \mathbf{X} \mathbf{1} = 1\}$. The constraint is due to the fact that $\mathbf{1}^\top \mathbf{T} = \mathbf{1}^\top$, $\mathrm{diag}(\boldsymbol{\pi})\mathbf{1} = \boldsymbol{\pi}$, and $\boldsymbol{\pi}^\top \mathbf{1} = 1$. Note that (22) is a standard constrained convex optimization problem that can be solved with off-the-shelf tools, such as CVX [40]. Having obtained $\hat{\mathbf{A}}$ from (22), we can then estimate the transition matrix as

$$\hat{\mathbf{T}} = \hat{\mathbf{A}}(\mathrm{diag}(\hat{\boldsymbol{\pi}}))^{-1}. \quad (23)$$

Note here that explicit knowledge of $\boldsymbol{\pi}$ is not required, as its estimate can be recovered from $\hat{\mathbf{A}}$ as

$$\hat{\boldsymbol{\pi}}^\top = \mathbf{1}^\top \hat{\mathbf{A}} = \mathbf{1}^\top \hat{\mathbf{T}} \mathrm{diag}(\hat{\boldsymbol{\pi}}) = \mathbf{1}^\top \mathrm{diag}(\hat{\boldsymbol{\pi}}).$$

The following proposition argues the consistency of the transition matrix estimates $\hat{\mathbf{T}}$.

Proposition 1. *Given accurate estimates of $\{\Gamma_m\}$ and $\boldsymbol{\pi}$, the estimate $\hat{\mathbf{T}}$ given by (22) and (23) approaches $\mathbf{T}$ as $N \to \infty$.*

Proof. By the law of large numbers, $\tilde{\mathbf{S}}_{mm'} \to \tilde{\mathbf{R}}_{mm'}$ as $N \to \infty$ for all m, m'. Since the objective in (22) is convex, from [41], we have that $\hat{\mathbf{A}}$ will converge to $\mathbf{A} = \mathbf{T}\mathrm{diag}(\boldsymbol{\pi})$ as $N \to \infty$. Finally, as $\hat{\mathbf{T}}$ can be recovered from $\hat{\mathbf{A}}$ in closed form [cf. (23)], the proof is complete. □

With estimates of $\{\hat{\Gamma}_m\}$, $\hat{\boldsymbol{\pi}}$ and $\hat{\mathbf{T}}$ at hand, estimates of the labels $\{y_n\}_{n=1}^{N}$ can be obtained using the method described in Section 4.1. Futhermore, the estimates of $\{\hat{\Gamma}_m\}$, $\hat{\boldsymbol{\pi}}$ and $\hat{\mathbf{T}}$ can be used to initialize an EM algorithm (a.k.a. Baum-Welch), whose details are provided in the next subsection.

Remark 1. *While here we employed the algorithm of [18] to estimate $\{\Gamma_m\}$, any other unsupervised ensemble classification algorithm, such as [14], [16], can be utilized too. In addition, methods that jointly estimate confusion matrices and Markov chain parameters such as [42], can also be appropriately modified for the ensemble classification task.*

4.3 EM Algorithm for Sequential Data

As with the i.i.d. case of Section 3, the EM algorithm developed here iteratively maximizes the log-likelihood of the observed learner responses. In order to update the parameters of interest $\theta := vec([\mathbf{T}, \Gamma_1, \dots, \Gamma_M])$ per iteration, the following two quantities have to be found

$$q_{nk} = \Pr(y_n = k|\mathbf{F}, \theta), \quad (24)$$

and

$$\xi_n(k, k') = \Pr(y_n = k, y_{n+1} = k' | \mathbf{F}, \boldsymbol{\theta}) . \tag{25}$$

Luckily, due to the causal structure of $\Pr(\boldsymbol{y})$, the aforementioned quantities can be estimated efficiently using the forward-backward algorithm [37], whose details can be found in Appendix A of the supplementary material, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TKDE.2020.3046645.

Algorithm 1. EM Algorithm for Sequential Data

Input: Learner responses $\{f_m(x_n)\}_{m=1,n=1}^{M,N}$, initial estimates $\mathbf{T}^{(0)}, \{\boldsymbol{\Gamma}_m^{(0)}\}_{m=1}^M$.
Output: Estimates $\hat{\mathbf{T}}, \{\hat{\boldsymbol{\Gamma}}_m\}_{m=1}^M$.
1: **while** not converged **do**
2: Estimate $\hat{q}_{nk}^{(i+1)}$ and $\hat{\xi}_n^{(i+1)}(k, k')$ using the forward-backward algorithm (Appendix A).
3: Estimate $\{\hat{\boldsymbol{\Gamma}}_m^{(i+1)}\}_{m=1}^M$ via (27).
4: Estimate $\hat{\mathbf{T}}^{(i+1)}$ via (26).
5: $i \leftarrow i + 1$
6: **end while**

Algorithm 2. Unsupervised Ensemble Classifier for Sequential Data

---

Input: Learner responses $\{f_m(x_n)\}_{m=1,n=1}^{M,N}$.
Output: Estimates of data labels $\{\hat{y}_n\}_{n=1}^N$.
1: Estimate $\boldsymbol{\pi}, \{\boldsymbol{\Gamma}_m\}_{m=1}^M$ via (17).
2: Estimate $\hat{\mathbf{T}}$ via (22) and (23).
3: Estimate $\hat{y}_n$ using the Viterbi algorithm [cf. Section 4.1].
4: If needed refine estimates of $\hat{\mathbf{T}}, \{\hat{\boldsymbol{\Gamma}}_m\}$ and $\{\hat{y}_n\}$ using Algorithm 1.

At iteration i, after obtaining $q_{nk}^{(i+1)}, \xi_n^{(i+1)}(k, k')$ for $k, k' = 1, \ldots, K$ and $n = 1, \ldots, N$, via the forward-backward algorithm, the transition and confusion matrix estimates can be updated as

$$[\hat{\mathbf{T}}^{(i+1)}]_{k'k} = \frac{\sum_{n=1}^{N-1} \xi_n^{(i+1)}(k', k)}{\sum_{n=1}^{N-1} q_{nk'}^{(i+1)}} \tag{26}$$

$$[\hat{\boldsymbol{\Gamma}}_m^{(i+1)}]_{k'k} = \frac{\sum_{n=1}^{N} q_{nk}^{(i+1)} \mathcal{I}(f_m(x_n) = k')}{\sum_{k''=1}^K \sum_{n=1}^N q_{nk}^{(i+1)} \mathcal{I}(f_m(x_n) = k'')} . \tag{27}$$

The EM iterations for sequential data with dependent labels is summarized in Algorithm 1, while the overall ensemble classifier for sequential data is tabulated in Algorithm 2. Note that the EM algorithm of this subsection does not rely on As3.

5 NETWORK DEPENDENT DATA

To tackle the networked data case, this section will introduce our novel approach to unsupervised ensemble classification of networked data. Given a graph $\mathcal{G}$ encoding data dependencies, recall from Section 2 that the joint pmf of all labels follows an MRF; thus, $\Pr(\boldsymbol{y}) = \frac{1}{Z}\exp(-U(\boldsymbol{y}))$ with

$$U(\boldsymbol{y}) = \frac{1}{2} \sum_{(n,n') \in \mathcal{E}} V(y_n, y_{n'}),$$

where $V(y_n, y_{n'})$ is the clique potential of the (n, n')th edge. Here, we select the clique potential as

$$V(y_n, y_{n'}) := \begin{cases} 0 & \text{if } y_n = y_{n'} \\ \delta_{nn'} & \text{if } y_n \neq y_{n'} \end{cases}, \tag{28}$$

where $\delta_{nn'} > 0$ is some predefined scalar, which controls how much we trust the given graph $\mathcal{G}$. The local energy at node (datum) n of the graph is then defined as

$$U_n(y_n) = \frac{1}{2} \sum_{n' \in \mathcal{N}_n} V(y_n, y_{n'}). \tag{29}$$

This particular choice of clique potentials forces neighboring nodes (data) of the graph to have similar labels, and has been successfully used in image segmentation [43], [44]. Under As1 and the aforementioned model, the joint pmf of label y_n and corresponding learner responses $\{f_m(x_n)\}_{m=1}^M$ given the neighborhood $\boldsymbol{y}_{\mathcal{N}_n}$ of node n, can be expressed as

$$
\begin{aligned}
&\Pr\Big(\{f_m(x_n)\}_{m=1}^M, y_n = k | \boldsymbol{y}_{\mathcal{N}_n} = \boldsymbol{k}_{\mathcal{N}_n}\Big) \\
&= \prod_{m=1}^M \Gamma_m(f_m(x_n), k) \Pr(y_n = k | \boldsymbol{y}_{\mathcal{N}_n} = \boldsymbol{k}_{\mathcal{N}_n}),
\end{aligned}
\tag{30}
$$

and accordingly the posterior probability of label y_n as

$$
\begin{aligned}
&\Pr\Big(y_n = k | \{f_m(x_n)\}_{m=1}^M, \boldsymbol{y}_{\mathcal{N}_n} = \boldsymbol{k}_{\mathcal{N}_n}\Big) \\
&\propto \prod_{m=1}^M \Gamma_m(f_m(x_n), k) \Pr(y_n = k | \boldsymbol{y}_{\mathcal{N}_n} = \boldsymbol{k}_{\mathcal{N}_n}) \\
&= \exp\left(-U_n(k) + \sum_{m=1}^M \log \Gamma_m(f_m(x_n), k)\right).
\end{aligned}
\tag{31}
$$

5.1 Label Estimation for Networked Data

Finding ML estimates of the labels $\hat{\boldsymbol{y}}$, under the aforementioned model, involves the following optimization problem:

$$
\begin{aligned}
\hat{\boldsymbol{y}} &= \arg\max_{\boldsymbol{y}} \Pr(\mathbf{F}, \boldsymbol{y}) = \arg\max_{\boldsymbol{y}} \Pr(\mathbf{F}|\boldsymbol{y}) \Pr(\boldsymbol{y}) \\
&= \arg\max_{\boldsymbol{y}} \frac{1}{Z} \exp(-U(\boldsymbol{y})) \Pr(\mathbf{F}|\boldsymbol{y}).
\end{aligned}
\tag{32}
$$

Unfortunately, (32) is intractable even for relatively small N, due to the structure of (5). This motivates well approximation techniques to obtain estimates of the labels.

Popular approximation methods include Gibbs sampling [45] and mean-field approximations [43]. Here, we opted for an iterative method called iterated conditional modes (ICM), which has been used successfully in image segmentation [44]. Per ICM iteration, we are given estimates $\{\hat{\boldsymbol{\Gamma}}_m\}_{m=1}^M$, and update the label of datum n by finding the k maximizing its local posterior probability; that is

$$
\begin{aligned}
\tilde{y}_n^{(t)} &= \arg\max_{k \in \{1,\ldots,K\}} \Pr\Big(y_n = k | \{f_m(x_n)\}_{m=1}^M, \tilde{\boldsymbol{y}}_{\mathcal{N}_n}^{(t-1)}\Big) \\
&= \arg\min_{k \in \{1,\ldots,K\}} U_n(k) - \sum_{m=1}^M \log\Big(\hat{\Gamma}_m(f_m(x_n), k)\Big),
\end{aligned}
\tag{33}
$$

where the superscript denotes the iteration index, $\tilde{y}_{\mathcal{N}_n}$ denotes the label estimates provided by the previous ICM iteration, and the second equality is due to (31). The optimization in (33) is carried out for $n = 1, \ldots, N$ until the values of $\tilde{y}$ have converged or until a maximum number of iterations $T_{\max}$ has been reached.

The next subsection puts forth an EM algorithm for estimating $\{\hat{y}_n\}_{n=1}^N$ and $\{\hat{\boldsymbol{\Gamma}}_m\}_{m=1}^M$.

5.2 EM Algorithm for Networked Data

As with the i.i.d. case in Section 3 and the sequential case in Section 4, the EM algorithm of this section seeks to iteratively maximize the marginal log-likelihood of observed learner responses. However, the Q-function [cf. Section 3.1] is now cumbersome to compute under the MRF constraint on $\boldsymbol{y}$.

For this reason, we resort to the approximation technique of the previous subsection to compute estimates of $q_{nk} = \Pr(y_n = k | \{f_m(x_n)\}_{m=1}^M; \boldsymbol{\theta})$. Specifically, per EM iteration i, we let $\hat{\boldsymbol{y}}^{(i)} := [\hat{y}_1^{(i)}, \ldots, \hat{y}_N^{(i)}]$ denote the estimates obtained by the iterative procedure of Section 5.1. Then, estimates $\hat{q}_{nk}^{(i+1)}$ are obtained as [cf. 31]

$$\hat{q}_{nk}^{(i+1)} = \frac{1}{Z'} \exp\left(-U_n^{(i+1)}(k) + \sum_{m=1}^M \log\left(\hat{\boldsymbol{\Gamma}}_m^{(i)}(f_m(x_n), k)\right)\right),$$
(34)

where

$$Z' = \sum_k \exp\left(-U_n^{(i+1)}(k) + \sum_{m=1}^M \log\left(\hat{\boldsymbol{\Gamma}}_m^{(i)}(f_m(x_n), k)\right)\right),$$

is the normalization constant, and $U_n^{(i+1)}(k)$ is given by

$$U_n^{(i+1)}(k) = \frac{1}{2} \sum_{n' \in \mathcal{N}_n} V(k, \hat{y}_{n'}^{(i+1)}).$$
(35)

Finally, the M-step that involves finding estimates of $\{\boldsymbol{\Gamma}_m\}_{m=1}^M$ is identical to the M-step of the EM algorithm of Section 3.1 for i.i.d. data; that is

$$[\hat{\boldsymbol{\Gamma}}_m^{(i+1)}]_{k'k} = \frac{\sum_{n=1}^N \hat{q}_{nk}^{(i+1)} \mathcal{I}(f_m(x_n) = k')}{\sum_{k''=1}^K \sum_{n=1}^N \hat{q}_{nk}^{(i+1)} \mathcal{I}(f_m(x_n) = k'')}.$$
(36)

Algorithm 3. EM Algorithm for Networked Data

Input: Learner responses $\{f_m(x_n)\}_{m=1,n=1}^{M,N}$, initial $\boldsymbol{y}^{(0)}$, $\{\boldsymbol{\Gamma}_m^{(0)}\}_{m=1}^M$, Data graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$.
Output: Estimates of data labels $\{\hat{y}_n\}_{n=1}^N$.
1: **while** not converged **do**
2:   **while** not converged AND $t < T_{\max}$ **do**
3:     **for** $n = 1, \ldots, N$ **do**
4:       Update $\tilde{y}_n^{(t)}$ using (33).
5:     **end for**
6:     $t \leftarrow t + 1$
7:   **end while**
8:   Compute $\hat{q}_{nk}^{(i+1)}$ using (34).
9:   Compute $\{\boldsymbol{\Gamma}_m^{(i+1)}\}_{m=1}^M$ using (36).
10:   $i \leftarrow i + 1$
11: **end while**

Similar to the i.i.d. case, the aforementioned EM solver deals with a non-convex problem. In addition, the ICM method outlined in Section 5.1 is a deterministic approach that performs greedy optimization. Therefore, proper initialization is crucial for obtaining accurate estimates of the labels and learner confusion matrices.

As with the decoupling approach of Section 4, here we first obtain estimates of learner confusion matrices $\{\hat{\boldsymbol{\Gamma}}_m\}_{m=1}^M$ and labels $\hat{\boldsymbol{y}}$, using the moment-matching algorithm of Section 3.2. These values are then provided as initialization to Algorithm 3. In cases where N is small to have accurate moment estimates, majority voting can be used instead to initialize Algorithm 3. The entire procedure for unsupervised ensemble classification with networked data is tabulated in Algorithm 4.

The next section will evaluate the performance of our proposed schemes.

Algorithm 4. Unsupervised Ensemble Classifier for Networked Data

Input: Learner responses $\{f_m(x_n)\}_{m=1,n=1}^{M,N}$, Data graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$
Output: Estimates of data labels $\{\hat{y}_n\}_{n=1}^N$
1: Estimate initial values of $\{\boldsymbol{\Gamma}_m\}_{m=1}^M$ via (17).
2: Estimate initial values of $\{\hat{y}_n\}_{n=1}^N$ using (18).
3: Refine estimates of $\{\hat{y}_n\}_{n=1}^N$ and $\{\hat{\boldsymbol{\Gamma}}_m\}_{m=1}^M$ using Algorithm 3.

Remark 2. *Contemporary Bayesian inference tools, such as variational inference [29], can also be appropriately modified to estimate labels of networked data that are expected to increase classification performance.*

6 NUMERICAL TESTS

The performance of the novel algorithms for both sequential and networked data is evaluated in this section using synthetic and real datasets. To showcase the importance of accounting for data dependencies in the unsupervised ensemble task, the proposed algorithms are compared to their counterparts designed for i.i.d. data. Since most of the numerical tests involve data with multiple, and potentially imbalanced classes, unless otherwise noted, the metrics evaluated are the per-class precision, per-class recall and the overall F-score [46]. F-score is the harmonic mean of precision and recall, that is

$$\text{F-score} = \frac{2}{K} \sum_{k=1}^K \frac{\text{Precision}_k * \text{Recall}_k}{\text{Precision}_k + \text{Recall}_k},$$
(37)

where $\text{Precision}_k, \text{Recall}_k$ denote the per-class precision and recall, respectively. Precision_k for a class k measures the proportion of the data predicted to be in class k that are actually from this class. Recall_k for a class k on the other hand measures the proportion of data that were actually in class k and were predicted to be in class k. To assess how accurately the algorithms can recover learner parameters, the average confusion matrix estimation error is also evaluated on synthetic data, as

$$\bar{\varepsilon}_{CM} := \frac{1}{M} \sum_{m=1}^M \frac{\|\boldsymbol{\Gamma}_m - \hat{\boldsymbol{\Gamma}}_m\|_1}{\|\boldsymbol{\Gamma}_m\|_1} = \frac{1}{M} \sum_{m=1}^M \|\boldsymbol{\Gamma}_m - \hat{\boldsymbol{\Gamma}}_m\|_1.$$
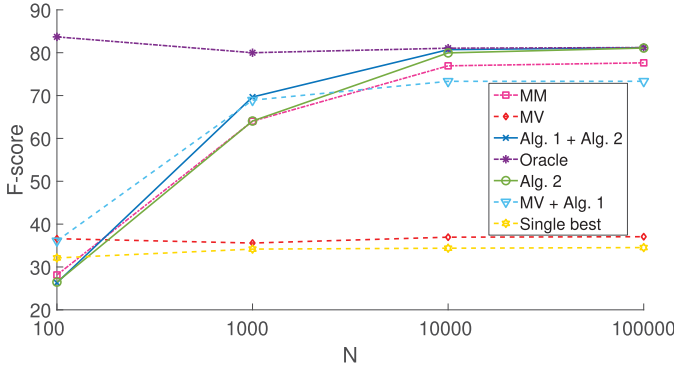(38)

Fig. 4. Average F-score for a synthetic sequential dataset with $K = 4$ and $M = 10$ learners.

All results represent averages over 10 independent Monte Carlo runs, using MATLAB [47]. Vertical lines in some figures indicate standard deviation.
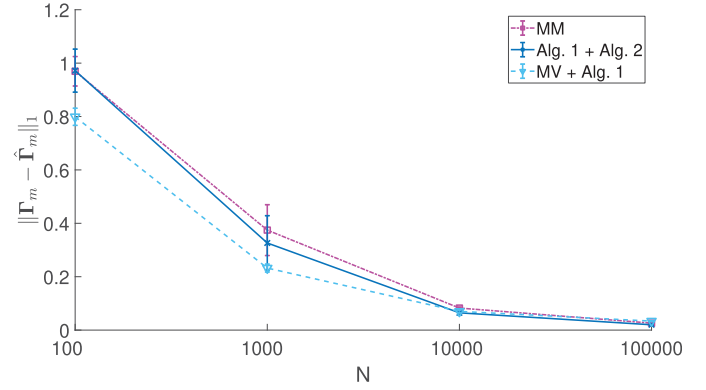
6.1 Sequential Data

For sequential data, Algorithm 2 with and without EM refinement (denoted as *Alg. 2 + Alg. 1* and *Alg. 2*, respectively) is compared to the single best classifier, with respect to F-score, of the ensemble (denoted as *Single best*); majority voting (denoted as *MV*); the moment-matching method of [18] described in Section 3.2 (denoted as *MM*); Algorithm 1 initialized with majority voting (denoted as *MV + Alg. 1*); and, "oracle" classifiers. "Oracle" classifiers solve (19) using Viterbi's algorithm [38], and have access to ground-truth learner confusion matrices $\{\boldsymbol{\Gamma}_m\}_{m=1}^M$ and the ground-truth Markov chain transition matrix $\mathbf{T}$. These "oracle" classifiers are used as an ideal benchmark for all other methods. The transition matrix estimation error $\|\mathbf{T} - \hat{\mathbf{T}}\|_1$ is also evaluated using synthetic data. For real data tests, instead of *MM* the EM algorithm of [14] initialized with *MM* is evaluated (denoted as *DS*.)

All datasets in this subsection are split into sequences. Here, we assume that per dataset these sequences are drawn from the same ensemble HMM [cf. Section 4]. The reported F-score represents the averaged F-score from all sequences.

6.1.1 Synthetic Data

For synthetic tests, S sequences of $N_s, s = 1, \ldots S$, ground-truth labels each, were generated from a Markov chain, whose transition matrix was drawn at random such that $\mathbf{T} \in \mathcal{C}$. Each of the $N = \sum_s N_s$ ground-truth labels $\{y_n\}_{n=1}^N$ corresponds to one out of K possible classes. Afterwards, $\{\boldsymbol{\Gamma}_m\}_{m=1}^M$ were generated at random, such that $\boldsymbol{\Gamma}_m \in \mathcal{C}$, for all $m = 1, \ldots, M$, and $\lfloor M/2 \rfloor + 1$ learners are better than random, as per As2. Then learners' responses were generated as follows: if $y_n = k$, then the response of learner m will be generated randomly according to the kth column of its confusion matrix, $\boldsymbol{\gamma}_{m,k}$ [cf. Section 2], that is $f_m(x_n) \sim \boldsymbol{\gamma}_{m,k}$.

Fig. 4 shows the average F-score for a synthetic dataset with $K = 4$, $M = 10$ learners and a variable number of data N and $N_s = 40$ for all $s = 1, \ldots, S$. Fig. 5 shows the average confusion and transition matrix estimation errors for varying N. As the number of data N increases the performance of the proposed methods approaches the performance of

(a) Confusion matrix estimation error
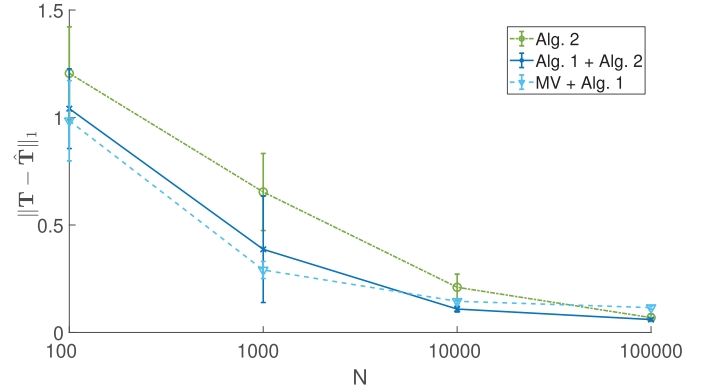
(b) Transition matrix estimation error

Fig. 5. Average estimation errors of confusion matrices and prior probabilities for a synthetic sequential dataset with $K = 4$ and $M = 10$ learners.

the "oracle" one. Accordingly, the confusion and transition matrix estimates are approaching the true ones as N increases. This is to be expected, as noted in [18], since the estimated moments are more accurate for large N. Interestingly, Algorithm 1 performs well when initialized with majority voting, even though it reaches a performance plateau as N increases. For small N however, it outperforms the other proposed methods. This suggests that initializing Algorithm 1 with majority voting is preferable when N is not large enough to obtain accurate moment estimation.

The next experiment evaluates the influence of the number of learners M for the sequential classification task. Figs. 6 and 7 showcase results for an experiment with $K = 4$, fixed number of data $N = 10^3, N_s = 40$ and a varying number of learners M. Clearly, the presence of multiple learners is beneficial, as the F-score increases for all algorithms, while the confusion and transition matrix errors decrease. As with the previous experiment, the performance of Alg. 1 + Alg. 2 improves in terms of F-score, as M increases.

6.1.2 Real Data

Further tests were conducted on three real datasets, the part-of-speech (POS) tagging dataset, the named entity recognition (NER) dataset, and the biomedical information extraction (IE) [30] dataset.

For the POS dataset $M = 10$ classifiers were trained using NLTK [48] on subsets of the Brown corpus [49] to provide
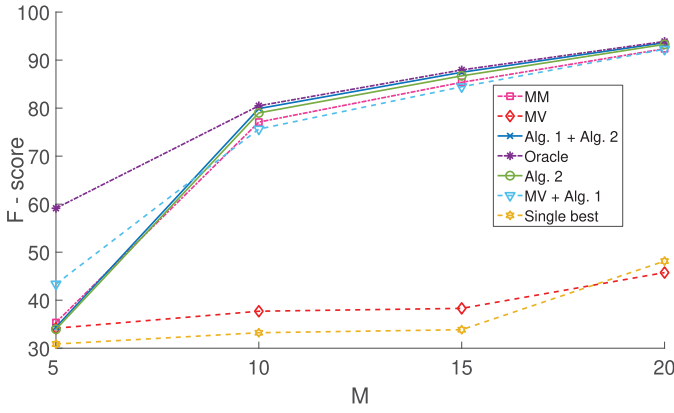
Fig. 6. Average F-score for a synthetic sequential dataset with $K = 4$ and $N = 10^3$ data.

(a) Confusion matrix estimation error

(b) Transition matrix estimation error

Fig. 7. Average estimation errors of confusion matrices and prior probabilities for a synthetic sequential dataset with $K = 2$ and $M = 10$ learners.

part-of-speech (POS) tags of text. The number of tags is $K = 12$. Then the classifiers provided POS tags for all words in the Penn Treebank corpus [50], which contains $N = 100,676$ words.

The NER dataset consists of 5,916 crowd annotated sentences from the CoNLL database [28], [51]. The dataset contains $N = 78,107$ words belonging to $K = 9$ distinct classes, each describing a different named-entity such as person, location, and organization. Corresponding to learners in our models, $M = 47$ human annotators provided labels for the words in the dataset.

The Biomedical IE dataset consists of 5,000 medical paper Abstracts, on which $M = 312$ human annotators, were tasked with marking all text spans in a given Abstract that identify the population of a randomized controlled trial. The dataset consists of $N = 7,880,254$ words belonging into $K = 2$ classes: in a span identifying the population or outside. For this particular dataset we evaluate Precision and Recall per sequence in the following way, which was suggested in [30]
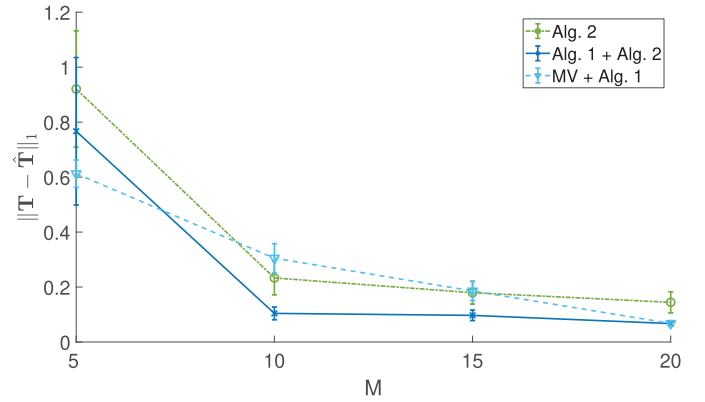
$$\text{Precision} = \frac{\text{\# true positive words}}{\text{\# words in a predicted span}}$$
$$\text{Recall} = \frac{\text{\# words in a predicted span}}{\text{\# words in ground-truth span}}.$$

These new definitions of precision and recall, allow us to credit partial matches. The justification for using these alternative definitions is that the previous ones are too strict for this task, where annotated sequences are especially long. We used the $M = 120$ annotators that had provided the

largest number of responses to mantain reasonable computational complexity. Results for all datasets are listed in Table 1. For the POS dataset, it can be seen that Alg. 1 + Alg. 2 performs best in all metrics. Similar results are showcased for the NER dataset, with majority voting achieving the best precision, and Alg. 1 + Alg. 2 exhibiting the best recall and overall F-score. For the Biomedical IE dataset, while majority voting achieves the best precision of all algorithms, due to its low recall, the overall F-score is low. However, Alg. 1 + Alg. 2 outperforms competing alternatives with regards to F-score, while MV + Alg. 1 exhibits the best recall. Note that the single best learners for the NER and Biomedical IE datasets are evaluated only on the subsets of data for which they have provided responses; the best learner for the NER dataset annotated approximately 12,500

TABLE 1
Results for Real Data Experiments With Sequential Data

Dataset	K	M	N	Metric	Single best	MV	DS	Alg. 2	Alg. 1 + Alg. 2	MV + Alg. 1
POS	12	10	100,676	Precision	0.2316	0.22916	0.23406	0.24785	**0.25856**	0.22972
				Recall	0.2573	0.23518	0.25629	0.24396	**0.26638**	0.24497
				F-score	0.2356	0.22598	0.22264	0.23352	**0.24735**	0.23007
NER	9	47	78,107	Precision	0.9*	**0.79**	0.77	0.74	0.77	0.75
				Recall	0.24*	0.59	0.66	0.89	**0.69**	0.66
				F-score	0.89*	0.68	0.71	0.62	**0.72**	0.70
Biomedical IE	2	120	7,880,254	Precision	0.94*	**0.89**	0.81	0.75	0.69	0.62
				Recall	0.76*	0.45	0.57	0.60	0.68	**0.74**
				F-score	0.84*	0.6	0.66	0.67	**0.68**	0.67

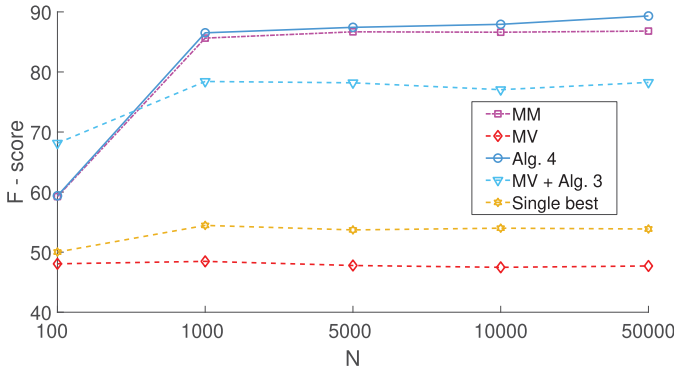*The asterisk ∗ indicates that results are from a subset of available data.*

Fig. 8. Average F-score for a synthetic networked dataset with $K = 4$, $M = 10$ learners and average degree $\bar{d} = 0.5$.

Fig. 9. Average estimation errors of confusion matrices for a synthetic networked dataset with $K = 4$ and $M = 10$ learners.

words, whereas the best learner for the Biomedical IE dataset annotated approximately 14,300 words. The performance of the proposed label aggregation methods relies on multiple parameters such as the number and ability of learners, and how well the proposed model approximates the learner behavior and data. The modest performance gains of Algorithms 2, 1 and MV + 1 compared to the single best learner may be attributed to such issues.

6.2 Networked Data

For networked data, Algorithm 4 (denoted as *Alg. 4*) is compared to the single best classifier, with respect to F-score, of the ensemble (denoted as *Single best*), majority voting (denoted as *MV*), the moment-matching method of [18] described in Section 3.2 (denoted as *MM*) and Algorithm 3 initialized with majority voting (denoted as *MV + Alg. 3*). For real data tests, instead of *MM* the EM algorithm of [14] initialized with *MM* is evaluated (denoted as *DS.*) The average degree $\bar{d}$ of the network is used to quantify the degree of data dependency ($\bar{d}$ is the number of connections averaged across nodes).

6.2.1 Synthetic Data

For the synthetic data tests, an N-node, K community graph is generated using a stochastic block model [52]. Each community corresponds to a class, and the labels $\{y_n\}_{n=1}^N$ indicate the community each node belongs to, i.e., $y_n = k$ if node n belongs to the kth community. Afterwards, $\{\mathbf{\Gamma}_m\}_{m=1}^M$ were generated at random, such that $\mathbf{\Gamma}_m \in \mathcal{C}$, for all $m = 1, \ldots, M$, and learners are better than random, as per As2. Then learners' responses were generated as follows: if $y_n = k$, then the response of learner m will be generated randomly according to the kth column of its confusion matrix, $\boldsymbol{\gamma}_{m,k}$ [cf. Section 2], that is $f_m(x_n) \sim \boldsymbol{\gamma}_{m,k}$. For the synthetic data tests, we set $\delta_{nn'} = M$. Fig. 8 shows the average F-score for a synthetic dataset with $K = 4$ and $M = 10$ learners for varying number of data N. Here the average degree is $\bar{d} = 0.5$. Fig. 9 shows the average confusion estimation error as N increases. As with sequential data, the F-score of the proposed algorithms increases with N growing, and confusion matrix estimation error decreases. *MV+ Alg. 3* quickly reaches a plateau of performance as *MV* also does not improve with increasing N. At the same time *Alg. 4* capitalizes on the initialization provided by *MM*. Fig. 10 shows the
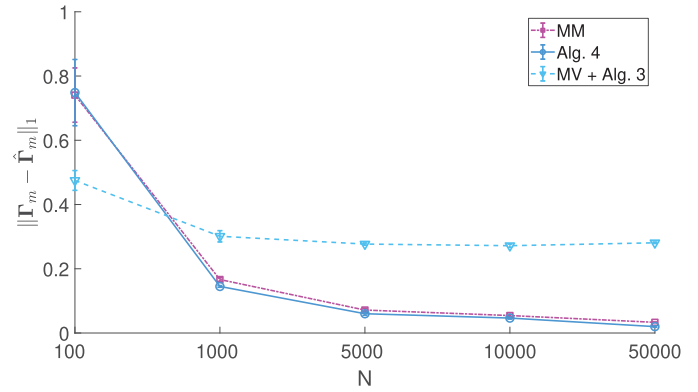
F-score for a similar experiment, but with network average degree $\bar{d} = 5$, i.e., higher graph connectivity. Here, we observe algorithmic behavior similar to that of the previous experiment; however, due to the higher connectivity of the graph *Alg. 4* has a greater F-score gap to *MM*. This indicates that networked data with higher connectivity benefit more from *Alg. 4* and *MV+ Alg. 3*.

6.2.2 Real Data

Further tests were conducted on six real datasets. For the Cora, Citeseer [23] and Pubmed [24] datasets the graph $\mathcal{G}$ and data features $\{x_n\}$ are provided with the dataset. In these cases, $M = 10$ classification algorithms from MATLAB's machine learning toolbox were trained on different randomly selected subsets of the datasets. Afterwards, these algorithms provided labels for all data in the dataset. Boston University's biomedical image library (BU-BIL) [53] magnetic resonance imaging (MRI) dataset contains 35 images of rabbit aortas with size 25×24. Learner responses per pixel are gathered through Amazon's mechanical turk. Here, the per-image graph $\mathcal{G}$ is the grid graph defined by the pixels of each image, that is each pixel is connected to its adjacent 8 pixels. For these datasets, we set $\delta_{nn'} = M$. The Music genre and Sentence Polarity datasets [33] are crowdsourcing datasets, where the features $\{x_n\}$ are provided and learner responses $\mathbf{F}$ are gathered through crowdsourcing platforms. In these cases, the graphs were generated from the data features using k-nearest neighbors. Since the graphs are generated from
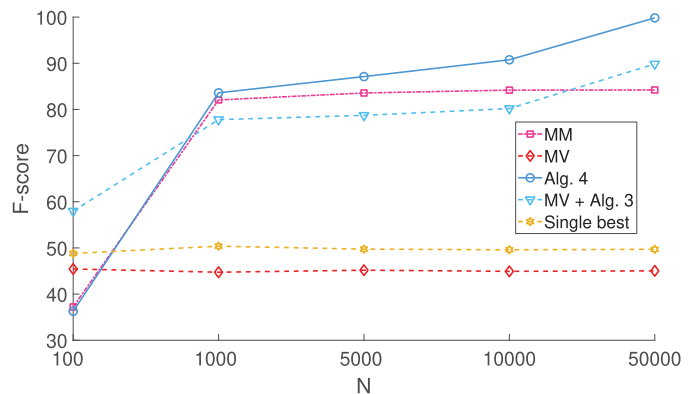

Fig. 10. Average F-score for a synthetic networked dataset with $K = 4$, $M = 10$ learners and average degree $\bar{d} = 5$.

TABLE 2
F-Score for Real Data Experiments With Networked Data

Dataset	K	M	N	$\bar{d}$	Single best	MV	DS	Alg. 4	MV + Alg. 3
Cora	10	10	2,708	3.9	0.51	0.2785	0.4228	**0.6412**	0.336
CiteSeer	7	10	3,312	2.77	0.45	0.4257	0.4449	**0.5244**	0.5224
Pubmed	3	10	19,717	4.49	0.717	0.6968	0.7437	**0.7667**	0.7595
Music Genre	10	44	700	4.8	1∗	0.7046	0.4746	0.7649	**0.8029**
Sen. Polarity	2	203	5,000	1.8	1∗	0.8895	0.9129	**0.9153**	0.9139
BU-BIL MRI	2	7	28,181	7.1	0.851	0.861	0.859	0.86	**0.863**

The asterisk ∗ indicates that results are from a subset of available data.

the data features, here we set $\delta_{nn'} = M_n/2$, where M_n is the number of learners that have provided a response for the nth datum.

The Cora, CiteSeer and Pubmed datasets are citation networks and the versions used here are preprocessed by [54]. The Cora dataset consists of $N = 2,708$ scientific publications classified into $K = 7$ classes. The features $\{x_n\}$ of this dataset are sparse 1,433-dimensional vectors and for this dataset each classification algorithm was trained on a random subset of 150 instances. The CiteSeer dataset consists of $N = 3,312$ scientific publications classified into one of $K = 6$ classes. The features $\{x_n\}$ of this dataset are sparse 3,703-dimensional vectors, and each classification algorithm was trained on a subset of 100 instances. The Pubmed dataset is a citation network that consists of $N = 19,717$ scientific publications from the Pubmed database pertaining to diabetes, classified into one of $K = 3$ classes. The features $\{x_n\}$ of this dataset are 500-dimensional vectors, and each classification algorithm was trained on a subset of 300 instances. Targeting segmentation, pixels of each image in the BU-BIL are classified in $K = 2$ classes, indicating whether they belong to a biological structure or not, by $M = 7$ human annotators. The total number of pixels from the 35 images is $N = 28,181$. The Music genre dataset contains $N = 700$ song samples (each of duration 30secs), belonging into $K = 10$ music categories, annotated by $M = 44$ human annotators. The sentence polarity dataset contains $N = 5,000$ sentences from movie reviews, classified into $K = 2$ categories (positive or negative), annotated by $M = 203$ human annotators.

The results for these datasets are listed in Table 2. In most datasets *Alg. 4* exhibits the best performance in terms of F-score followed closely by *MV+Alg. 3*. For the Music Genre and BU-BIL MRI datasets however, *MV+Alg. 3* outperforms *Alg. 4*. This is to be expected for the Music Genre dataset, as N is relatively small for this dataset and as such the estimated learner moments are not very accurate. This can also be seen from the fact that *MV* outperforms *DS*. Similarly, for the BU-BIL MRI dataset *MV* outperforms *DS*, explaining the better performance of *MV+Alg. 3*. For all datasets, *Alg. 4* and *MV+Alg. 3* consistently outperform the single best classifier. Also, note that the single best learners for the Music genre and sentence polarity datasets are evaluated only on the subsets of data for which they have provided responses. In particular, the best learner for the Music-genre has annotated 10 data, while the best learner for the Sentence-polarity has annotated only 6 data. Another interesting observation is that for most datasets having a relatively large average degree $\bar{d}$, *Alg. 4* and *MV+Alg. 3* have a greater performance

gap to their counterparts that do not account for the structure of networked data. Similar to synthetic data, this suggests that well connected datasets can benefit more from these types of approaches. Similarly to the sequential case, the modest performance gains of Algorithms 3, and MV + 3 compared to the single best learner in the BU-BIL MRI dataset may be attributed to modeling discrepancies or significantly different ability levels between the learners. All in all, these results show that inclusion of graph information can be beneficial for the unsupervised ensemble or crowdsourced classification task, even when the graph is noisy (as with Sentence Polarity or the Music genre datasets.)

7 CONCLUSIONS AND FUTURE DIRECTIONS

This paper introduced two novel approaches to unsupervised ensemble and crowdsourced classification in the presence of data dependencies. Two types of data dependencies were investigated: i) Sequential data; and ii) Networked data, where the dependencies are captured by a known graph. The performance of our novel schemes was evaluated on real and synthetic data.

Several interesting research avenues open up: i) Distributed and online implementations of the proposed algorithms; ii) use of contemporary tools such as variational inference to boost performance of the novel approaches; iii) ensemble classification with dependent classifiers and dependent data; iv) development of more realistic learner models for dependent data; v) extension of the proposed methods to semi-supervised ensemble learning; vi) rigorous performance analysis of the proposed models.

ACKNOWLEDGMENTS

REFERENCES

[1] T. G. Dietterich, "Ensemble methods in machine learning," in *Proc. Int. Workshop Multiple Classifier Syst.*, 2000, pp. 1–15.
[2] Z.-H. Zhou, *Ensemble Methods: Foundations and Algorithms*, 1st ed. London, U.K./Boca Raton, FL, USA: Chapman & Hall/CRC, 2012.
[3] L. I. Kuncheva, *Combining Pattern Classifiers: Methods and Algorithms*. Wiley-Interscience, USA, 2004.
[4] L. Rokach, "Ensemble-based classifiers," *Artif. Intell. Rev.*, vol. 33, pp. 1–39, Feb. 2010.
[5] Y. Freund *et al.*, "Experiments with a new boosting algorithm," in *Proc. Int. Conf. Mach. Learn.*, 1996, pp. 148–156.
[6] D. H. Wolpert, "Stacked generalization," *Neural Netw.*, vol. 5, no. 2, pp. 241–259, 1992. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0893608005800231

[7] P. Smyth and D. Wolpert, "Linearly combining density estimators via stacking," *Mach. Learn.*, vol. 36, no. 1, pp. 59–83, Jul. 1999. [Online]. Available: https://doi.org/10.1023/A:1007511322260

[8] F. Wright *et al.*, "Multidisciplinary cancer conferences: A systematic review and development of practice standards," *Eur. J. Cancer*, vol. 43, pp. 1002–1010, 2007.

[9] M. Micsinai, F. Parisi, F. Strino, P. Asp, B. D. Dynlacht, and Y. Kluger, "Picking ChIP-seq peak detectors for analyzing chromatin modification experiments," *Nucleic Acids Res.*, vol. 40, no. 9, May 2012, Art. no. e70.

[10] A. Timmermann, "Forecast combinations," in *Handbook of Economic Forecasting*, vol. 1. Amsterdam, The Netherlands: Elsevier, 2006, pp. 135–196.

[11] D. C. Brabham, "Crowdsourcing as a model for problem solving: An introduction and cases," *Convergence*, vol. 14, no. 1, pp. 75–90, 2008.

[12] A. Kittur, E. H. Chi, and B. Suh, "Crowdsourcing user studies with mechanical turk," in *Proc. SIGCHI Conf. Hum. Factors Comput. Syst.*, 2008, pp. 453–456.

[13] Figure8. Accessed: 2020. [Online]. Available: https://www.figure-eight.com/

[14] A. P. Dawid and A. M. Skene, "Maximum likelihood estimation of observer error-rates using the EM algorithm," *J. Roy. Statist. Soc. Ser. Appl. Statist.*, vol. 28, pp. 20–28, 1979.

[15] D. R. Karger, S. Oh, and D. Shah, "Efficient crowdsourcing for multi-class labeling," *ACM SIGMETRICS Perform. Eval. Rev.*, vol. 41, no. 1, pp. 81–92, 2013.

[16] Y. Zhang, X. Chen, D. Zhou, and M. I. Jordan, "Spectral methods meet EM: A provably optimal algorithm for crowdsourcing," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2014, pp. 1260–1268.

[17] A. Jaffe, B. Nadler, and Y. Kluger, "Estimating the accuracies of multiple classifiers without labeled data," in *Proc. 18th Int. Conf. Artif. Intell. Statist.*, 2015, Art. no. 4.

[18] P. A. Traganitis, A. Pagès-Zamora, and G. B. Giannakis, "Blind multiclass ensemble classification," *IEEE Trans. Signal Process.*, vol. 66, no. 18, pp. 4737–4752, Sep. 2018.

[19] P. A. Traganitis, A. Pagès-Zamora, and G. B. Giannakis, "Learning from unequally reliable blind ensembles of classifiers," in *Proc. 5th IEEE Global Conf. Signal Inf. Process.*, 2017, pp. 106–110.

[20] P. A. Traganitis and G. B. Giannakis, "Blind multi-class ensemble learning with dependent classifiers," in *Proc. 26th Eur. Signal Process. Conf.*, 2018, pp. 2025–2029.

[21] N. Nguyen and Y. Guo, "Comparisons of sequence labeling algorithms and extensions," in *Proc. 24th Int. Conf. Mach. Learn.*, 2007, pp. 681–688. [Online]. Available: http://doi.acm.org/10.1145/1273496.1273582

[22] N. D. Sidiropoulos, L. De Lathauwer, X. Fu, K. Huang, E. E. Papalexakis, and C. Faloutsos, "Tensor decomposition for signal processing and machine learning," *IEEE Trans. Signal Process.*, vol. 65, no. 13, pp. 3551–3582, Jul. 2017.

[23] Q. Lu and L. Getoor, "Link-based classification," in *Proc. 20th Int. Conf. Int. Conf. Mach. Learn.*, 2003, pp. 496–503. [Online]. Available: http://dl.acm.org/citation.cfm?id=3041838.3041901

[24] G. Namata, B. London, L. Getoor, and B. Huang, "Query-driven active surveying for collective classification," in *Proc. Int. Conf. Mach. Learn. Workshop Mining Learn Graphs*, 2012.

[25] J. M. Hammersley and P. E. Clifford, "Markov random fields on finite graphs and lattices," *Unpublished Manuscript*, 1971. [Online]. Available: http://www.statslab.cam.ac.uk/~grg/books/hammfest/hamm-cliff.pdf

[26] N. Dalvi, A. Dasgupta, R. Kumar, and V. Rastogi, "Aggregating crowdsourced binary ratings," in *Proc. Int. Conf. World Wide Web*, 2013, pp. 285–294.

[27] T. Bonald and R. Combes, "A minimax optimal algorithm for crowdsourcing," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2017, pp. 4352–4360. [Online]. Available: http://papers.nips.cc/paper/7022-a-minimax-optimal-algorithm-for-crowdsourcing.pdf

[28] F. Rodrigues, F. Pereira, and B. Ribeiro, "Sequence labeling with multiple annotators," *Mach. Learn.*, vol. 95, no. 2, pp. 165–181, May 2014. [Online]. Available: https://doi.org/10.1007/s10994–013-5411-2

[29] M. J. Beal, "Variational algorithms for approximate Bayesian inference," PhD dissertation, Gatsby Comput. Neurosci. Unit, Univ. College London, 2003.

[30] A. T. Nguyen, B. C. Wallace, J. J. Li, A. Nenkova, and M. Lease, "Aggregating and predicting sequence labels from crowd annotations," in *Proc. 55th Annu. Meeting Assoc. Comput. Linguistics*, 2017, pp. 299–309.

[31] Y. Yan, R. Rosales, G. Fung, R. Subramanian, and J. Dy, "Learning from multiple annotators with varying expertise," *Mach. Learn.*, vol. 95, no. 3, pp. 291–327, Jun. 2014. [Online]. Available: https://doi.org/10.1007/s10994–013-5412-1

[32] P. Ruiz, P. Morales- Álvarez, R. Molina, and A. Katsaggelos, "Learning from crowds with variational Gaussian processes," *Pattern Recognit.*, vol. 88, pp. 298–311, Apr. 2019. [Online]. Available: http://decsai.ugr.es/vip/files/journals/1-s2.0-S0031320318304060-main.pdf

[33] F. Rodrigues, F. Pereira, and B. Ribeiro, "Gaussian process classification and active learning with multiple annotators," in *Proc. 31st Int. Conf. Mach. Learn.*, 2014, pp. 433–441. [Online]. Available: http://proceedings.mlr.press/v32/rodrigues14.html

[34] P. A. Traganitis, "Blind ensemble classification of sequential data," in *Proc. IEEE Data Sci. Workshop*, 2019, pp. 130–134.

[35] S. M. Kay, *Fundamentals of Statistical Signal Processing: Estimation Theory*, vol. 1. Englewood Cliffs, NJ, USA: Prentice Hall, 1993.

[36] K. Huang, N. D. Sidiropoulos, and A. P. Liavas, "A flexible and efficient algorithmic framework for constrained matrix and tensor factorization," *IEEE Trans. Signal Process.*, vol. 64, no. 19, pp. 5052–5065, Oct. 2016.

[37] L. R. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proc. IEEE*, vol. 77, no. 2, pp. 257–286, Feb 1989.

[38] G. D. Forney, "The viterbi algorithm," *Proc. IEEE*, vol. 61, no. 3, pp. 268–278, Mar. 1973.

[39] A. Kontorovich, B. Nadler, and R. Weiss, "On learning parametric-output HMMs," in *Proc. 30th Int. Conf. Mach. Learn.*, 2013. [Online]. Available: http://proceedings.mlr.press/v28/kontorovich13.html

[40] M. Grant and S. Boyd, "CVX: Matlab software for disciplined convex programming, version 2.1," Mar. 2014. [Online]. Available: http://cvxr.com/cvx

[41] V. Vapnik, *The Nature of Statistical Learning Theory*. Berlin, Germany: Springer, 2013.

[42] K. Huang, X. Fu, and N. D. Sidiropoulos, "Learning hidden Markov models from pairwise co-occurrences with applications to topic modeling," in *Proc. 35th Int. Conf. Mach. Learn.*, 2018, pp. 2073–2082.

[43] J. Zhang, "The mean field theory in EM procedures for Markov random fields," *IEEE Trans. Signal Process.*, vol. 40, no. 10, pp. 2570–2583, Oct 1992.

[44] J. Besag, "On the statistical analysis of dirty pictures," *J. Roy. Statist. Soc. B*, vol. 48, no. 3, pp. 48–259, 1986.

[45] G. Casella and E. I. George, "Explaining the Gibbs sampler," *Amer. Statistician*, vol. 46, no. 3, pp. 167–174, 1992.

[46] D. M. W. Powers, "Evaluation: From precision, recall and F-measure to ROC, informedness, markedness and correlation," *J. Mach. Learn. Technol.*, vol. 2, no. 1, pp. 37–63, 2011.

[47] MATLAB, *version 8.6.0 (R2015b)*. Natick, Massachusetts: The MathWorks Inc., 2015.

[48] E. Loper and S. Bird, "NLTK: The natural language toolkit," in *Proc. ACL Workshop Effective Tools Methodologies Teaching Natural Lang. Process. Comput. Linguistics*, 2002, pp. 63–70. [Online]. Available: https://doi.org/10.3115/1118108.1118117

[49] W. N. Francis and H. Kucera, "Brown corpus manual," Dept. Linguistics, Brown Univ., Providence, RI, USA, 1979. [Online]. Available: http://icame.uib.no/brown/bcm.html

[50] M. Marcus *et al.*, "The penn treebank: Annotating predicate argument structure," in *Proc. Workshop Hum. Lang. Technol.*, 1994, pp. 114–119. [Online]. Available: https://doi.org/10.3115/1075812.1075835

[51] E. F. Tjong Kim Sang and F. De Meulder, "Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition," in *Proc. 7th Conf. Natural Lang. Learn. HLT-NAACL*, 2003, pp. 142–147. [Online]. Available: https://www.aclweb.org/anthology/W03–0419

[52] P. W. Holland, K. B. Laskey, and S. Leinhardt, "Stochastic blockmodels: First steps," *Soc. Netw.*, vol. 5, no. 2, pp. 109–137, 1983. [Online]. Available: http://www.sciencedirect.com/science/article/pii/0378873383900217

[53] D. Gurari *et al.*, "How to collect segmentations for biomedical images? A benchmark evaluating the performance of experts, crowdsourced non-experts, and algorithms," in *Proc. IEEE Winter Conf. Appl. Comput. Vis.*, 2015, pp. 1169–1176.

[54] D. Berberidis, A. N. Nikolakopoulos, and G. B. Giannakis, "Adaptive diffusions for scalable learning over graphs," *IEEE Trans. Signal Process.*, vol. 67, no. 5, pp. 1307–1321, Mar. 2019.

Panagiotis A. Traganitis (Member, IEEE) received the diploma degree in electrical and computer engineering from the National Technical University of Athens, Athens, Greece, in 2013, and the MSc and PhD degrees in electrical engineering from the University of Minnesota (UMN), Twin Cities, Minnesota, in 2015 and 2019, respectively. Currently, he is a postdoctoral researcher with the Department of Electrical and Computer Engineering (ECE), University of Minnesota, Twin Cities. His research interests include statistical signal processing, ensemble learning, distributed learning, big data analytics, and network science.

Georgios B. Giannakis (Fellow, IEEE) received the diploma degree in electrical engineering from the National Technical University of Athens, Athens, Greece, in 1981, and the MSc degree in electrical engineering, in 1983, the MSc degree in mathematics, in 1986, and the PhD degree in electrical engineering, in 1986, both from the University of Southern California (USC), Los Angeles, California, respectively. He was a faculty member with the University of Virginia from 1987 to 1998, and since 1999 he has been a professor with the University of Minnesota, where he holds an ADC endowed chair, a University of Minnesota McKnight presidential chair in ECE, and serves as director of the Digital Technology Center. His general interests span the areas of statistical learning, communications, and networking - subjects on which he has published more than 450 journal papers, 750 conference papers, 26 book chapters, two edited books, and two research monographs. Current research focuses on data science, and network science with applications to the Internet of Things, brain, and power networks with renewables. He is the (co-) inventor of 33 patents issued, and the (co-) recipient of nine best journal paper awards from the IEEE Signal Processing (SP) and Communications Societies, including the G. Marconi Prize Paper Award in wireless communications. He also received the IEEE-SPS Nobert Wiener Society Award (2019); EURASIP's A. Papoulis Society Award (2020); Technical Achievement Awards from the IEEE-SPS (2000) and from EURASIP (2005); the IEEE ComSoc Education Award (2019); and the IEEE Fourier Technical Field Award (2015). He is a fellow of the National Academy of Inventors, the European Academy of Sciences, and EURASIP. He has served the IEEE in a number of posts, including that of a distinguished lecturer for the IEEE-SPS.

▷ **For more information on this or any other computing topic, please visit our Digital Library at** www.computer.org/csdl.