Multilabel classification with multivariate time series predictors

Yuezhang Che[1], Yunzhang Zhu[2], and Xiaotong Shen[3]

Last updated August 31, 2019

***Abstract*—In many applications, multilabel classification involves time-series predictors, as in multilabel video classification. How to account for the temporal dependencies with respect to input variables remains an issue, especially in action learning from videos. Motivated by the problem of video categorization and captioning, we propose a nonlinear multilabel classifier based on a hidden Markov model and a weighted loss separating false positive and negative classification errors. This allows us to account for label dependence and temporal dependencies of input variables in classification. Computationally, we derive a decomposable algorithm based on block-wise coordinate descent for non-convex minimization, where it permits not only to block-wise updates but also label-wise updates, leading to scalable computation. Theoretically, we derive the Bayes rule and prove that the proposed method consistently recovers the optimal performance of the Bayes rule. In simulations, the proposed method compares favorably with its competitors ignoring either label dependence or time-dependence. Finally, the utility of the proposed method is demonstrated by an application to *ActivityNet Captions dataset* for understanding a video's contents.**

***Index Terms*—Label dependence, hidden Markov models, video sequence, nonconvex minimization, scalability.**

I. Introduction

DIGITAL information processing has become an essential part of modern life. Given the explosive growth of information in the big data era, it is extremely critical that digital information is accurately summarized and organized. One important yet challenging problem is the automatic summarization of video content, having enormous applications in video advertisements, online video searching and browsing, the automatic recommendation on movies based on personal preference, essentially any electronic commerce platform. In this article, we investigate statistical modeling of multilabel classification of multivariate time-series, motivated from video categorization associated with captioning.

Multilabel classification is useful in modern applications, including protein function classification [1], music categorization [2], and semantic scene classification [3], [4]. Multilabel classification permits multiple labels to be assigned to each instance. There is a large body of literature on multilabel classification. Relevant references include binary relevance [5]–[8], a co-occurrence-based method [9], Markov/Bayesian/dependence network methods [10], and pseudo-likelihood based on an Ising model [11]. Whereas the latter approaches intend to account for label dependence, working loss functions such as the Hamming loss [12] and the Jaccard distance [13] can not account for label dependence. However, most existing methods assume independent learning samples, yet difficult to treat predictors with temporal dependencies such as time series. Two major challenges emerge. First, often semantic labels are used, particularly in video categorization, exhibiting strong label dependence

[1] School of Statistics and Management, Shanghai University of Finance and Economics; [2] Department of Statistics, The Ohio State University, Columbus, OH 43210; [3]School of Statistics, University of Minnesota, Minneapolis, MN 55455. The authors thank the editors, the associate editor and anonymous referees for helpful comments and suggestions. Research supported in part by NSF grants DMS-1415500, DMS-1712564, DMS-1721216, DMS-1712580, DMS-1721445, and DMS-1721445.

Manuscript received XXX; revised XXX.

due to word-to-word relations. For example, "football" and "tennis" are two semantic labels that are unconditionally dependent, usually co-occurring in a sports event, whereas they are conditionally independent given the equipment used in sports [14]. In such a situation, how to exploit the label dependence becomes critical to improve the classification performance. Second, how to treat predictors induced by piecewise stationary time series, such as a video sequence, remains open.

In this article, we will address these two challenges by developing novel classification models to account for label dependence as well as temporal dependencies. Particularly, we motivate our methodological development by the problem of understanding the content of a video and develop classifiers to recognize multiple concurrent actions of a video. For a video, multiple semantic labels describe a video's content given that each scene may involve multiple actions simultaneously. On the other hand, a video's temporal dependence may switch from one scene to another, which remains steady over certain consecutive frames before transiting to the next state. Indeed, action recognition has been under-studied in statistics, although it has received some attention in machine learning and the film industry. In [15], motions are combined with appearance features to learn to recognize, while [16] develops temporal segment neural networks. For single-label classification for videos, deep learning has shown some promising empirical results, including long-short memory fully convolutional networks (LSTM-FCN) and its attention version ALSTM-FCN [17]. Both LSTM-FCN and ALSTM-FCN use temporal convolutional blocks for feature extraction and a LSTM block for learning temporal dependencies. In [17], LSTM-FCN and ALSTM-FCN achieve the state of the art performance against strong competitors in some benchmarks for the activity or action recognition, including the distance-based methods [18], [19], traditional feature-based algorithms [20], classical machine learners [21], [22], dimensional reduction techniques [23]–[25], as well as other deep learners.

To address the two aforementioned challenges, we develop a nonlinear classifier for multilabel classification with multivariate time-series predictors, motivated by the unique characteristics of a video. Specifically, we model our nonlinear classifier that is driven by a hidden state of a hidden Markov model, which leads to predictions with a local piecewise constant structure. This is well-suited for semantic content prediction for videos, because semantic entities or actions in videos often remains to be unchanged until it switches to a different scene. The transition probabilities of the hidden states are estimated by the regularized maximum likelihood while a classifier based on the weighted loss is constructed given a hidden state. Moreover, we propose to use a nonlinear classifier based on random forests to account label dependence as well as a weighted classification loss separating false from negative positives. The resulting classifier integrates nonlinear classification with a Gaussian hidden Markov model, in which random forest is used to construct a nonlinear classifier.

Computationally, we propose a margin loss-based approach replacing the likelihood function for classification. To make computation scalable, we develop a strategy of divide-and-conquer to divide the corresponding optimization into many equivalent small ones to optimize, as opposed to the use of the EM algorithm. On this ground, we derive a decomposable algorithm based on blockwise coordinate decent [26] for convex and nonconvex minimizations. The block involving continuous parameters can be estimated using random forest and graphical lasso algorithm, while the block involving latent states can be estimated using Viterbi algorithm. Theoretically, we prove that the proposed classifier recovers the optimal performance of the Bayes rule consistently. To our knowledge, we are unaware of any results for this type of result with dependent features. One key challenge of our analysis is to show that consistent recovery of the proposed method remains intact even with a small fraction of samples whose hidden states are incorrectly estimated. Towards this end, we derive technical conditions under which the number of such cases can be small. Numerically, we demonstrate the utility of the proposed method in simulated and real examples, where it compares favorably with

its counterpart without accounting for temporal dependence [14], the random forests [27], and deep learners [17]. Moreover, in the benchmark *ActivityNet Captions dataset*, the proposed method performs well in terms of learning a video's actions. This demonstrates the advantages of modeling the temporal dependence in classification involving dependent inputs such as a video's frames.

The article is organized as follows. Section 2 briefly introduces the propose methodology. Section 3 provides theoretical justification of the proposed classifier in terms of recovering the optimal performance of the Bayes rule. Section 4 presents the numerical results on simulation study and real data analysis. Section 5 summarizes the conclusions and Appendix provides technical proofs.

II. Methodology

Consider a situation in which input-output pairs $\{\boldsymbol{X}_t, \boldsymbol{Y}_t\}_{t=1}^T$ are observed, where, at time t, $\boldsymbol{X_t} = \{X_t^{(1)}, \ldots, X_t^{(p)}\}^T \in \mathbb{R}^p$ is a multivariate time-series predictor vector and $\boldsymbol{Y}_t = (Y_t^1, \ldots, Y_t^K)^T$ is a label vector with each $Y_j^t \in \{\pm 1\}$. This occurs, for instance, in video categorization, in which $\boldsymbol{X}_t$ represents a video clip's feature vector observed at time t and $\boldsymbol{Y}_t$ encodes certain objects or actions at time t. In such a situation, it is known that $(\boldsymbol{X}_t, \boldsymbol{Y}_t)$ has a piecewise smooth temporal dependence over time t, or local stationary, as one action or objects may remain in many consecutive frames of a video. Our focus is predicting the label values of $\boldsymbol{Y}_{t_0}$ at a future observation time point $t = t_0$ based on $\{\boldsymbol{X}_t, \boldsymbol{Y}_t\}_{t=1}^T$.

To quantify the aforementioned temporal dependence of $\boldsymbol{X}_t$, we first introduce our hidden Markov models. Let Z_t be an unobserved latent state at time t, where $Z_t = s$; $s = 1, \ldots, S$, and S is a total number of possible latent states, which is a discrete Markov chain with an initial probability $P(Z_1 = s) = \pi_s$ and a transition probability $P(Z_t = s' | Z_{t-1} = s) = p_{ss'}$. Assume, without of loss of generality, that Z_t is stationary and irreducible. Given $Z_t = s$, $\boldsymbol{X}_t$ follows a multivariate Gaussian distribution $N(\boldsymbol{\mu}_s, \boldsymbol{\Omega}_s^{-1})$, with a mean $\boldsymbol{\mu}_s$ and a precision matrix $\boldsymbol{\Omega}_s$.

To predict the outcome of $\boldsymbol{Y}_t = (Y_t^1, \cdots, Y_t^K)^\top$, we introduce classification function vector $\boldsymbol{f}_s(\boldsymbol{x}_t) = (f_{s,1}(\boldsymbol{x}_t), \ldots, f_{s,K}(\boldsymbol{x}_t))^\top$ with $f_{s,k}(\boldsymbol{x}_t)$ for label k at latent state $Z_t = s$; $k = 1, \ldots, K$, mapping from $\boldsymbol{x}_t \in \mathbb{R}^p$ to $\{-1, 1\}^K$. For each k, Y_t^k is predicted by $\hat{Y}_t^k = \sum_s \mathbb{I}(Z_t = s) \operatorname{Sign}(f_{s,k}(\boldsymbol{X}_t))$, where $\mathbb{I}(\cdot)$ is the indicator function and $\operatorname{Sign}(\cdot)$ is the sign function. As suggested by Figure 1, temporal dependence is modeled through the latent state Z_t of $(\boldsymbol{Y}_t, \boldsymbol{X}_t)$, inducing a piecewise stationary time-series.

To measure the performance of $\boldsymbol{f}$, we define the generalization error, $\operatorname{Err}(\boldsymbol{f}) = \mathbb{E}L(\boldsymbol{Y}, \boldsymbol{f}) = \sum_{k=1}^K \mathbb{E}[\sum_{s=1}^S \mathbb{I}(Z_t = s) L_k(\boldsymbol{Y}_t, f_{s,k}(\boldsymbol{X}_t))]$, where L_k is the loss weighted multilabel classification loss in [14], which enables to account for partial correctness.

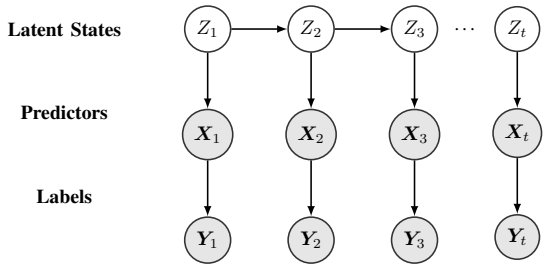

Fig. 1. Diagram of the information flow for the hidden Markov model.

A. Multilabel classification based on hidden Markov models

This section develops a nonlinear classifier accounting for time-dependence of $(\boldsymbol{Y}_t, \boldsymbol{X}_t)$ through a hidden Markov model with latent states defined by a discrete Markov chain Z_t. In this fashion, $(\boldsymbol{Y}_t, \boldsymbol{X}_t)$ are dependent, whereas $(\boldsymbol{Y}_t, \boldsymbol{X}_t)$ given Z_t can be conditionally independent; $t = 1, \ldots, T$. As a consequence, a higher accuracy of prediction can be realized than a classifier ignoring such a temporal structure.

Given observed input-output pair $\{\boldsymbol{X}_t, \boldsymbol{Y}_t\}_{t=1}^T$, the negative log-likelihood based on $\{\boldsymbol{Y}_t, \boldsymbol{X}_t\}_{t=1}^T$ given $\{Z_t = z_t\}_{t=1}^T$ can be written as

$$\begin{aligned} &-\textstyle\sum_{t=1}^T \log \mathbb{P}(\boldsymbol{Y}_t|\boldsymbol{X}_t, Z_t = z_t) \\ &-\textstyle\sum_{t=1}^T \log \mathbb{P}(\boldsymbol{X}_t|Z_t = z_t) \\ &-\textstyle\sum_{t=1}^{T-1} \log p_{z_t,z_{t+1}} - \log \pi_{z_1}, \end{aligned} \tag{1}$$

where $p_{z_t,z_{t+1}} = \mathbb{P}(Z_{t+1} = z_{t+1}|Z_t = z_t)$ denotes the transition probability from $Z_t = z_t$ to $Z_{t+1} = z_{t+1}$ with $1 \le z_t, z_{t+1} \le S$, and $\pi_{z_1} = \mathbb{P}(Z_1 = z_1)$ is an initial probability.

Motivated by the likelihood function in (1), we develop a loss-based method without specifying the likelihood for classification. Towards this end, we consider a weighted loss [14] separating false positive from negatives as follows:

$$L(\boldsymbol{Y}_t, \boldsymbol{f}_s) = \sum_{k=1}^K L_k(\boldsymbol{Y}_t, f_{s,k}(\boldsymbol{X}_t)), \tag{2}$$

where $L_k(\boldsymbol{Y}_t, f_{s,k}(\boldsymbol{X}_t)) = |\delta_k(\boldsymbol{Y}_t)|\mathbb{I}(\delta_k(\boldsymbol{Y}_t) f_{s,k}(\boldsymbol{X}_t) < 0)$, $\delta_k(\boldsymbol{Y}_t) = \sum_{j:Y_t^j=+1} \omega_{+jk} - \sum_{j:Y_t^j=-1} \omega_{-jk}$, and $\omega_{+jk} \ge 0$ and $\omega_{-jk} \ge 0$ are false positive and negative misclassification errors of Y_t^j by $f_{s,k}(\boldsymbol{X}_t)$. Loss (2) quantifies label dependence and accounts for partial label correctness. It is additive to each label, which makes scalable computation possible. In (1), we replace $-\log \mathbb{P}(\boldsymbol{Y_t}|\boldsymbol{X_t}, Z_t = z_t)$ by its surrogate loss $L(\cdot,\cdot)$ and re-write $\mathbb{P}(\boldsymbol{X}_t|Z_t = z_t)$ as $\mathbb{P}_{\boldsymbol{\mu}_{z_t},\boldsymbol{\Omega}_{z_t}}$. This renders the proposed cost function with respect to parameters $\boldsymbol{\theta} = \{\boldsymbol{f}, z_t, \boldsymbol{\mu}_s, \boldsymbol{\Omega}_s, \pi_s, p_{ss'}\}$ with $t = 1, \ldots, T$ and $1 \le s, s' \le S$:

$$\begin{aligned} &H(\boldsymbol{\theta}) = \\ &\sum_{t=1}^T L(\boldsymbol{Y}_t, \boldsymbol{f}_{z_t}(\boldsymbol{X}_t)) - \sum_{t=1}^T \log \mathbb{P}_{\boldsymbol{\mu}_{z_t},\boldsymbol{\Omega}_{z_t}}(\boldsymbol{X}_t) \\ &- \sum_{t=1}^{T-1} \log p_{z_t,z_{t+1}} - \log \pi_{z_1}, \end{aligned} \tag{3}$$

where $\log \mathbb{P}_{\boldsymbol{\mu}_{z_t},\boldsymbol{\Omega}_{z_t}}(\boldsymbol{X}_t) = -\frac{p}{2}\log(2\pi) + \frac{1}{2}\log|\boldsymbol{\Omega}_{z_t}| - \frac{1}{2}(\boldsymbol{X}_t - \boldsymbol{\mu}_{z_t})^T \boldsymbol{\Omega}_{z_t}(\boldsymbol{X}_t - \boldsymbol{\mu}_{z_t})$.

For a typical hidden Markov model, the cost function to minimize involves the integral of (3) over all possible latent states, which is often treated with the EM algorithm that is usually not scalable. Then we choose to optimize (3) with respect to the hidden states, which is computationally amenable. To deal with high-dimensional parameters, we regularize (3) using a graph Lasso (GLasso) penalty [28] on $\boldsymbol{\Omega}_s$ and consider the following minimization

$$\min_{\boldsymbol{\theta}=(\boldsymbol{f},z_t,\mu_s,\boldsymbol{\Omega}_s,\pi_s,p_{ss'})} H(\boldsymbol{\theta}) + \lambda \sum_{s=1}^S J(\boldsymbol{\Omega}_s) \tag{4}$$

where $\lambda > 0$ is a tuning parameter and $J(\boldsymbol{\Omega}_s) = \sum_{i<j} |\Omega_{s,(i,j)}|$ is the GLasso penalty, which encourages each precision matrix $\boldsymbol{\Omega}_s$ to be sparse. Moreover, we propose to use random forest to treat the minimization of $\sum_{t=1}^T L(\boldsymbol{Y}_t, \boldsymbol{f}_{z_t}(\boldsymbol{X}_t))$ over nonlinear function space $\boldsymbol{f}_{z_t}$. One benefit of using a nonlinear classifier is that it can effectively model label dependence, c.f., [14].

B. Computation

To make the proposed method (4) scalable, we utilize the strategy of divide-and-conquer to solve two minimizations separately, which is equivalent to minimize (4). Particularly, we derive a bi-block coordinate descent algorithm with involving a latent variable block $\{Z_t\}_{t=1}^T$ and a parameter block $(f_{s,k}, \boldsymbol{\mu_s}, \boldsymbol{\Omega_s}, \pi_s, p_{s's})_{1\le s',s\le S,1\le k\le K}$, to solve (6) iteratively until convergence, while alternatively between these two blocks. Given the present solution $(\hat{f}_{s,k}, \hat{\boldsymbol{\mu}}_s, \hat{\boldsymbol{\Omega}}_s, \hat{\pi}_s, \hat{p}_{s's})$ for the continuous parameter, we employ *Viterbi algorithm* [29], a dynamic programming algorithm, solving (3) with respect to $\{Z_t = z_t\}_{t=1}^T$, which amounts to solving

$$\begin{aligned} \min_{z_1,\ldots,z_T} -\Big(&\textstyle\sum_{t=1}^T \log \mathbb{P}_{\hat{\boldsymbol{\mu}}_{z_t},\hat{\boldsymbol{\Omega}}_{z_t}}(\boldsymbol{X}_t) \\ &-\textstyle\sum_{t=1}^{T-1} \log \hat{p}_{z_t,z_{t+1}} - \log \hat{\pi}_{z_1}\Big), \end{aligned} \tag{5}$$

to yield the estimated latent state $\{\hat{Z}_t = \hat{z}_t\}_{t=1}^T$. Then given the current solution of latent states $\{\hat{Z}_t = \hat{z}_t\}_{t=1}^T$, solving (3) with respect to

$$\Big\{ f_{z_t,k}, \boldsymbol{\mu_{z_t}}, \boldsymbol{\Omega_{z_t}}, \pi_{z_t}, p_{z_t,z_{t+1}}; \\ 1 \le z_t, z_{t+1} \le S, 1 \le k \le K \Big\},$$

which is again decomposed into three minimization subproblems subsequently:

$$\min_{f_{s,k}} \sum_{\substack{t=1\\ \hat{z}_t=s}}^{T} \sum_{k=1}^{K} L_k(\boldsymbol{Y}_t, f_{s,k}(\boldsymbol{X}_t)); \quad s=1,\ldots,S, k=1,\ldots,K, \tag{6}$$

$$\min_{\mu_s,\Omega_s} -\sum_{\substack{t=1\\ \hat{z}_t=s}}^{T} \log \mathbb{P}_{\boldsymbol{\mu}_s,\boldsymbol{\Omega}_s}(\boldsymbol{X}_t) + \lambda J_2(\boldsymbol{\Omega}_s); \quad l=1,\ldots,S, \tag{7}$$

$$\min_{\substack{p_{s's},\pi_l\\ \sum_{s'} p_{s's}=1 \sum_s \pi_s=1}} -\sum_{s',s} N_{s's} \log p_{s's} - \log \pi_s; \quad s',s=1,\ldots,S. \tag{8}$$

To solve (6), we decompose it into K separate subproblems to solve utilizing the additive property of $L_k(\cdot,\cdot)$:

$$\min_{f_{s,k}} \sum_{\substack{t=1\\ \hat{z}_t=s}}^{T} L_k(\boldsymbol{Y}_t, f_{s,k}(\boldsymbol{X}_t)) = \min_{f_{s,k}} \sum_{\substack{t=1\\ \hat{z}_t=s}}^{T} |\delta_k(\boldsymbol{Y}_t)| \mathbb{I}(\delta_k(\boldsymbol{Y}_t) f_{s,k}(\boldsymbol{X}_t) < 0); \tag{9}$$

$k=1,\ldots,K$, permitting parallel computation. As a result, the computational complexity becomes linear in K, where we employ a weighted version of random capability of variable selection. In (9), we solve weighted binary classification for predicting $\mathrm{Sign}(\delta_k(\boldsymbol{Y}_t))$, where the overall misclassification loss is weighted by $\delta_k(\boldsymbol{Y}_t)$. A fast algorithm [30] can be modified with weights for estimating classifier $f_{s,k}$ which is a random forests.

To solve (7), note that $\log \mathbb{P}_{\boldsymbol{\mu}_{z_t},\boldsymbol{\Omega}_{z_t}}(\boldsymbol{X}_t) = -\frac{p}{2}\log(2\pi) + \frac{1}{2}\log|\boldsymbol{\Omega}_{z_t}| - \frac{1}{2}(\boldsymbol{X}_t - \boldsymbol{\mu}_{z_t})^T \boldsymbol{\Omega}_{z_t}(\boldsymbol{X}_t - \boldsymbol{\mu}_{z_t})$. Then (7) reduces to

$$\min_{\boldsymbol{\mu}_s,\boldsymbol{\Omega}_s} \quad \frac{1}{2} \sum_{\substack{t=1\\ \hat{z}_t=s}}^{T} (\boldsymbol{X}_t - \boldsymbol{\mu}_s)^T \boldsymbol{\Omega}_s (\boldsymbol{X}_t - \boldsymbol{\mu}_s) - \log|\boldsymbol{\Omega}_s| + \lambda J(\boldsymbol{\Omega}_s).$$

Then the sample mean vector and covariance matrix can be written as

$$\hat{\boldsymbol{\mu}}_s = \frac{1}{N_{s\cdot}} \sum_{\substack{t=1\\ \hat{z}_t=s}}^{T} \boldsymbol{X}_t, \quad \widehat{\boldsymbol{\Sigma}}_s = \frac{1}{N_{s\cdot}-1} \sum_{\substack{t=1\\ \hat{z}_t=s}}^{T} (\boldsymbol{X}_t - \boldsymbol{\mu})(\boldsymbol{X}_t - \boldsymbol{\mu})^\top. \tag{10}$$

Given a L_1-penalty $J(\boldsymbol{\Omega}) = \sum_{i\neq j} |\Omega_{i,j}|$ and $\lambda > 0$, the L_1-regularized Gaussian maximum likelihood estimate of $\boldsymbol{\Omega}_s$ is the solution of the following problem:

$$\underset{\Omega \succ 0}{\mathrm{argmin}} -\log\det \boldsymbol{\Omega}_s + \mathrm{Tr}(\widehat{\boldsymbol{\Sigma}}_s \boldsymbol{\Omega}_s) + \lambda \sum_{i\neq j} |\Omega_{s,(i,j)}|, \tag{11}$$

which reduces to a GLasso problem that can be efficiently solved [28], [31], where $\Omega_{s,(i,j)}$ denotes the (i,j)th entry of $\boldsymbol{\Omega}_s$.

To solve (8), note that $\hat{p}_{ss'} = \frac{N_{ss'}}{N_{s\cdot}}$, where $N_{ss'} = \#\{t : \hat{Z}_t = s \text{ and } \hat{Z}_{t+1} = s'\}$, $N_{s\cdot} = \sum_{s'=1}^{S} N_{ss'}$, and $\hat{\pi}_s$ is 1 if $Z_1 = s$ and otherwise 0.

The optimization strategy for computing the solution of (4) is summarized in Algorithm 1.

Algorithm 1

Step 1: Specify initial values for $(\hat{\boldsymbol{\mu}}_s^{(0)}, \hat{\boldsymbol{\Omega}}_s^{(0)}, \hat{\pi}_s^{(0)}, \hat{p}_{s's}^{(0)})$ based on sample statistics of each cluster generated by the k-means algorithm;

Start iteration:

Step 2: Estimate latent states $z_1, \cdots, z_T$ using the *Viterbi algorithm* and via (5);

Step 3: Given updated latent states, minimize (9) for $f_{s,k}$; $k = 1, \cdots, K$, $s = 1, \cdots, S$, (11) for $\boldsymbol{\Omega}_s$; $s = 1, \ldots, S$, and update $\hat{p}_{ss'} = \frac{N_{ss'}}{N_{s\cdot}}$;

Step 4: Terminate when the amount of improvement in terms of the cost function is smaller than the pre-specified tolerance.

III. Statistical learning theory

This section develops a learning theory to investigate the proposed method's capability of recovery of the Bayes performance, as measured by the generalization error. In particular, we establish consistency result for a global minimizer of (4) $\hat{f}$, where the generalization error of the classifier defined by $\hat{\boldsymbol{f}}$ is

$$\mathrm{Err}(\hat{\boldsymbol{f}}) = \sum_k \mathbb{E} L_k(\hat{f}_k) = \sum_k \sum_s P(Z=s)\mathbb{E}(L_k(\hat{f}_{s,k})|Z=s),$$

where $L_k = \sum_s \mathbb{I}(Z = s)L_k(\hat{f}_{s,k})$, $L_k(\hat{f}_{s,k})$ is the loss for label k at state s, and $\mathbb{E}$ and $\mathbb{E}(\cdot|Z = s)$ are the expectation and the conditional expectation given $Z = s$, respectively. Similarly, the Bayes error is $\mathrm{Err}(\boldsymbol{f}^\star) = \sum_{k=1}^K \sum_{s=1}^S P(Z = s)E(L_k(f^\star_{s,k})|Z = s)$, where $f^\star_{s,k}$ denotes the Bayes decision rule for label k at state s. Consistency of classifier $\hat{\boldsymbol{f}}$ means that $\mathrm{Err}(\hat{\boldsymbol{f}}) \to \mathrm{Err}(\boldsymbol{f}^\star)$ as the sample size tends to ∞.

Lemma 1 below gives the Bayes decision rule $f^\star_{s,k}$ that minimizes the generalization error $\mathrm{Err}(f_{s,k}) = \mathbb{E}(L_k(f_{s,k})|Z = s)$.

Lemma 1: (Bayes decision rule) Given loss L_k, the Bayes decision function $f^\star_{s,k}$ is expressed as

$$f^\star_{s,k}(\boldsymbol{x}_t) = \mathrm{Sign}(\mathbb{E}(\delta_k(\boldsymbol{Y}_t)|\boldsymbol{X}_t = \boldsymbol{x}_t, Z_t = s)); \quad (12)$$

$k = 1, \ldots, K; s = 1, \ldots, S$, where $\mathbb{E}(\delta_k(\boldsymbol{Y}_t)|\boldsymbol{X}_t = \boldsymbol{x}_t, Z_t = s) = \sum_{j=1}^K (\omega_{-jk}\mathbb{P}_s(Y_j = 1|\boldsymbol{X}_t = \boldsymbol{x}_t, Z_t = s) - \omega_{+jk}\mathbb{P}_s(Y_j = -1|\boldsymbol{X}_t = \boldsymbol{x}_t, Z_t = s))$.

Next we analyze our random forests classifier f in (2), which is a voting classifier of simple decision trees considered in [14], [32], [33]. Specifically, for a single tree, a coordinate of $\boldsymbol{X} = (X_1, \ldots, X_p)$ is chosen at each node with each X_r having a probability $p_r \in (0, 1)$ of being selected, and the selected partition is split along the randomly chosen variable at the midpoint. Then a voting classifier $\hat{f}^{(B)}_{s,k}$ for label k at state s is defined as the average of B independent tree classifiers

$$\hat{f}^{(B)}_{s,k} = B^{-1} \sum_{b=1}^B \hat{f}_{s,kb}(\boldsymbol{x}), k = 1, \ldots, K \quad (13)$$

where $\hat{f}^{(b)}_{s,k}$; $b = 1, \ldots, B$, are independent single-tree classifiers with the same number of variable splits. Note that this split scheme differ from that used by the standard random forest in how features and split are chosen, yet it is more amenable to theoretical analysis. As a technical note, [14] examines the same voting classifier for a weighted loss function with independent observations. Our analysis may be regarded as an expansion of the theory in [14] to a multivariate time series model generated by a hidden Markov model.

Two key components of our theoretical analysis are as follows. First, we prove that the fraction of time points at which latent states can be incorrectly estimated is essentially small. Second, even with incorrectly estimated latent states, the classifier given the estimated latent states remains consistent when the number of splits can be adjusted.

The following assumptions are imposed for consistent consistency.

Assumption 1: (Model assumptions)

(A1) (Parameter space) The parameter space $\Theta = (\boldsymbol{\mu_s}, \boldsymbol{\Omega_s}, \pi_s, p_{s's})_{1 \le s', s \le S}$ is compact and the true parameter value $\theta^\star$ is an interior point of Θ. The dimension of the feature dimension p does not grow with T. Moreover, for all $T > 1$, $\theta^\star$ is identifiable in that it is a unique solution of $\mathbb{E}[\log \frac{p_{\theta^\star}(\boldsymbol{X})}{p_\theta(\boldsymbol{X})}] = 0$ for $\theta \in \Theta$.

(A2) (Transition probabilities) For the ss' element $p_{ss'}$ of the transition probability matrix, there exist constants p_+ and p_- such that $0 < p_- \le p_{ss'} \le p_+ < 1$; $s, s' = 1, \cdots, S$.

(A3) (Eigenvalues) There exist constants $0 < \eta_l \le \eta_u < \infty$ such that $\eta_l \le \min \lambda_{l,s} \le \max \lambda_{u,s} \le \eta_u$, where $\lambda_{l,s}$ and $\lambda_{u,s}$ are the smallest and largest eigenvalues of the covariance matrix $\boldsymbol{\Omega}_s^{-1}$.

Conditions (A1)–(A3) in Assumption 1 are the same as A2.1, A3.1 to A3.4 in Section 3 of [34] for multivariate hidden Markov mixture models. These conditions ensure that the all continuous parameters in the hidden Markov model can be estimated consistently.

Assumption 2: (Transition probabilities) Assume that

$$(S-1)\exp\left(-\frac{\min_{s \ne s'} h^2_{ss'}}{2}\right) \max_{s,s',s'' \ne s} \sqrt{\frac{p_{ss''}}{p_{ss'}}} < 1\,. \quad (14)$$

where $h_{ss'}$ denotes the Hellinger-distance between two normal densities $N(\mu_s, \Omega_s^{-1})$ and $N(\mu_{s'}, \Omega_{s'}^{-1})$.

Assumption 2 concerns the transition matrix and the degree of separation between the mixtures. The conditions are imposed so that the latent states in the hidden Markov model can mostly be estimated consistently.

Assumption 3: (Relation of tree tuning and transition probabilities) Let M be the number of splits for each individual tree in the random forest. For $\boldsymbol{X}_t \in \mathcal{X}_s$ at state s, the sample size $n_s \to \infty$ as

$T = \sum_{s=1}^{S} n_s \to \infty$; $s = 1, \ldots, S$. Moreover, the transition probability $p_{ss'}$, the number of splits M, the feature sampling probability p_r, and the number of categories K, satisfy

$$M \to \infty, \quad M \max_{s \neq s'} \sqrt{\frac{p_{s's}}{p_{s's'}}} \to 0,$$
$$\frac{K^3 M}{n_s} \to 0, \quad p_r \log M - 2\log(K) \to \infty\,;$$

$r = 1, \ldots, p$; $s = 1, \ldots, S$, as $T \to \infty$.

Assumption 3 requires that $\max_{s \neq s'} \frac{p_{s's}}{p_{s's'}} \to 0$, which says that the probability of staying at the same state as the previous one is much larger than that of moving to a different state. Moreover, if K is fixed and all variables are equally likely to be chosen for splitting, that is, p_r remains the same across r, then a sufficient condition for Assumption 3 are simplified as

$$\max_{s \neq s'} \sqrt{\frac{p_{s's}}{p_{s's'}}} \to 0 \tag{15}$$

when choosing M to be, for example, $M = (\max_{s \neq s'} \sqrt{\frac{p_{s's}}{p_{s's'}}})^{-1/2}$.

Assumption 4: (Smooth Bayes decision functions) Assume that the Bayes decision functions $f^{\star}_{s,k}$; $k = 1, \ldots, K$; $s = 1, \ldots, S$, are uniformly L-Lipschitz continuous in that $\max_{1 \leq k \leq K} |f^{\star}_{s,k}(\boldsymbol{x}_1) - f^{\star}_{s,k}(\boldsymbol{x}_2)| \leq D||\boldsymbol{x}_1 - \boldsymbol{x}_2||_2$, where $D > 0$ is a constant independent of p, K, n_s and $\|\cdot\|_2$ denotes the L_2-norm.

Assumption 4 is usually required for analyzing a tree voting classifier [32].

Theorem 1 establishes consistent recovery of the Bayes performance by the proposed method under the loss (3).

Theorem 1: Under Assumptions 1–4, the classifier defined by a global minimizer of (4) $\hat{\boldsymbol{f}}$ is consistent as $T \to \infty$.

IV. Numerical examples

This section investigates operating characteristics of the proposed method with respect to the impacts of (1) latent states, (2) different covariance matrix structures of input vector $\boldsymbol{X}_t$, (3) overlapping input vector $\boldsymbol{X}_t$, and (4) label dependence of components of $\boldsymbol{Y}$. Importantly, it is compared against the random forest with and without assuming latent states to be known, referred to as random forests (RF) and Oracle.

A. Simulation

1) Model Setting: Our example concerns three latent states with $T = 10000$, where the temporal dependence is induced by a Markov chain with initial probabilities $\pi = (1/3, 1/3, 1/3)$ and a transition matrix

$$P = \begin{pmatrix} 0.4 & 0.5 & 0.1 \\ 0.2 & 0.3 & 0.5 \\ 0.4 & 0.3 & 0.3 \end{pmatrix}.$$

To generate a multivariate time-series, a $p = 100$-dimensional input vector $\boldsymbol{X}_t$ is drawn from a normal distribution $N(\boldsymbol{\mu}_l, \boldsymbol{\Omega}_l^{-1})$ given that $Z_t = l$; $l = 1, \cdots, 3$. Two situations are considered: overlapping and non-overlapping samples between states, by varying the relative size of $\boldsymbol{\mu}_l$ and $\boldsymbol{\Omega}_l$. In the non-overlapping situation, the input vector $\boldsymbol{X}_t$ is perfectly separable with respect to different states, which is in contrast to the overlapping case. Specifically, let $\boldsymbol{\mu}_l = l_1 \boldsymbol{\mu}_{\mathbf{0}}$ and $\boldsymbol{\Omega}_l = l_2 \boldsymbol{\Omega}^{(i)}$ given state $Z_t = l$, where $\boldsymbol{\mu}_0 = (1, 1, 1)$ and $\boldsymbol{\Omega}^{(i)}$ is to be specified. Here $l_1 = -3, 0, 3$ and $l_2 = 3, 2, 3$ yield the overlapping situation, while $l_1 = -12, 0, 12$ and $l_2 = 3, 2, 3$ give the non-overlapping situation. To investigate the influence of covariance matrix on the performance, we consider the following three covariance structures, as in [35]: $\boldsymbol{\Sigma}^{(1)}$ with entries $\sigma_{ij}^{(1)} = \rho^{|i-j|}$ with $\rho = 0.7$, $\boldsymbol{\Sigma}^{(2)}$ with entries $\sigma_{ij}^{(2)} = \boldsymbol{I}(|i-j| = 0) + 0.4\boldsymbol{I}(|i-j| = 1) + 0.2\boldsymbol{I}(|i-j| = 2) + 0.2\boldsymbol{I}(|i-j| = 3) + 0.1\boldsymbol{I}(|i-j| = 4)$, and $\boldsymbol{\Sigma}^{(3)}$ with entries $\sigma_{ij}^{(3)} = \boldsymbol{I}(i = j) + 0.5\boldsymbol{I}(i \neq j)$, where $\boldsymbol{\Sigma}^{(i)} = [\boldsymbol{\Omega}^{(i)}]^{-1}$.

To investigate the impact of modeling label dependence, in each state, we use an Ising model defined in (16) as in [14], modeling linear pairwise conditional dependencies. Given $\boldsymbol{Y} = (Y_1, \cdots, Y_K)$ follows a conditional Ising model given $\boldsymbol{X}_t$ and latent state Z_t. The probability density of $\boldsymbol{Y}$ given $\boldsymbol{X}_t = \boldsymbol{x}_t$ and $Z_t = l$ is

$$P(\boldsymbol{y}|\boldsymbol{x}, z = l) = \frac{1}{M(\boldsymbol{\alpha}_l(\boldsymbol{x}))} \exp\Big(\frac{1}{2}\sum_{j=1}^{K} \alpha_{l,jj}(\boldsymbol{x}) y_j + \frac{1}{2}\sum_{m>j} \alpha_{l,jm}(\boldsymbol{x}) y_j y_m\Big) \tag{16}$$

where $\boldsymbol{y} = (y_1, \cdots, y_K)^T$,$\boldsymbol{\alpha}_l(\boldsymbol{x}) = (\alpha_{l,11}(\boldsymbol{x}), \alpha_{l,12}(\boldsymbol{x}), \cdots, \alpha_{l,pp}(\boldsymbol{x}))$ is a $p(p+1)/2$-dimensional vector and $M(\boldsymbol{\alpha}_l(\boldsymbol{x}))$ is a partition function to ensure that 2^p probabilities summing up to one. Now we let $\alpha_{l,jm} = \alpha_{l,mj} = \boldsymbol{x}^T\boldsymbol{\theta}_{l,jm} + \theta_{l,jm0}$, $m < j$, where $\boldsymbol{\theta}_{l,jm} = (\theta_{l,jm1}, \cdots, \theta_{1,jmp})$. $\theta_{l,jmu}$, $u = 1, \cdots, p$ are drawn from Unif$(-\gamma_1, \gamma_1)$, and $\theta_{l,jm0}$ is drawn from Unif$(-\gamma_2, \gamma_2)$, for different states, $\gamma_1 = 10, 50, 20$ and $\gamma_2 = -10, 50, -5$. In total, we generate 30 labels.

Nine methods are compared, including the proposed method (Our), two binary relevance methods based on linear (LSVM-BR) and Gaussian kernel (NSVM-BR) support vector machines both ignoring label dependence with estimated hidden states, two standard methods based on linear (LSVM) and Gaussian kernel (NSVM-BR) support vector machines ignoring both label and temporal dependencies, the weighted random forests (RF) [14] ignoring temporal dependence with hidden states sharing the identical mean and covariance, the oracle method (Oracle) with known hidden states, and two deep learning methods using temporal convolution and long short memory network for temporal dependence without attention mechanism (LSTM-FCN) and with attention mechanism (ALSTM-FCN) [17]. For deep learning, we use the publicly available code and default parameters provided by the authors. The training is processed on a single GTX 1050 Ti GPU.

The accuracy of classification is evaluated on a test sample under the weighted loss (9) and the Hamming loss [12]. The following two weight matrices are used for the weighted loss in (9):

$$W_+ = \boldsymbol{I}_{K\times K} \text{ and}$$
$$W_- = \begin{pmatrix} .6 & \frac{.4}{K-1} & \cdots & \frac{.4}{K-1} \\ \frac{.4}{K-1} & .6 & \cdots & \frac{.4}{K-1} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{.4}{K-1} & \frac{.4}{K-1} & \cdots & .6 \end{pmatrix},$$

where $\boldsymbol{I}$ is the identity matrix and each row or column sum of W_- is one.

2) Numerical results: In the case of $K = 5$ involving $S = 3$ hidden states, as shown in Table I, the proposed method is the best performer or close to the best performer in terms of the weighted loss, which is close to NSVM-BR and the optimal performance—the oracle. Interestingly, it outperforms the other competitors RF by about 3% to 11% and substantially outperforms LSVM-BR, LSVM and NSVM by about 38% to 50%. The proposed method also performs far better than the deep learning methods LSTM-FCN and ALSTM-FCN, improving the weighted loss by about 70%. In general, nonlinear methods performs better than linear methods, and methods that account for label dependence performs better than those that do not. Moreover, it is noted that the proposed method performs better in the separable case, which is anticipated as the latent states can be estimated more precisely in this situation. Overall, the improved performance of the proposed method can be attributed to latent state modeling.

In the cases of $K = 20$ and $K = 40$, the proposed method is either the best performer or close to the best, particularly achieving a similar performance of the oracle method (Oracle). The amounts of improvement of the proposed method over LSVM-BR, NSVM-BR, LSVM, NSVM, RF, LSTM-FCN, and ALSTM-FCN range from 45% to 92%, from 4% to 10%, from 53% to 97%, from 36% to 60%, from 0% to 7%, from 182% to 284%, and from 180% to 287%. Roughly, large improvements occur for challenging situations, as K increases. As in the case of $K = 5$, the first seven methods all outperform the last two deep learning methods, which is attributed to the fact that the deep learning methods could neither estimate the explicit temporal structure nor account for the label dependence. Moreover, the generalization error escalates about 5% to 60% when $\boldsymbol{X}_t$ has a more complex covariance structure such as $\boldsymbol{\Omega}^{(2)}$ and $\boldsymbol{\Omega}^{(3)}$ as opposed to $\boldsymbol{\Omega}^{(1)}$.

The proposed method continues to fare well even under the other commonly used loss—Hamming loss, as indicated in Table II. In summary, the proposed method performs well even when the classification loss differs from the evaluation loss in this case. This could be attributed to the fact that label dependence has been adequately taken into

TABLE I
TEST ERRORS (STANDARD ERRORS IN PARENTHESES) IN ORDER OF 10^{-4} OF VARIOUS METHODS UNDER THE WEIGHTED LOSS OVER 100 SIMULATIONS. TYPES 1 AND 2 DENOTE NON-SEPARABLE AND SEPARABLE CASES AND $\Omega^{(1)}$–$\Omega^{(3)}$ REPRESENT THREE TEMPORAL DEPENDENCIES.

K	Type	Ω	Our	LSVM-BR	NSVM-BR	LSVM	NSVM	RF	LSTM-FCN	ALSTM-FCN	Oracle
5	1	$\Omega^{(1)}$	916(11)	1609(23)	913(10)	1680(14)	1468(13)	945(11)	3349(64)	3231(73)	917(10)
		$\Omega^{(2)}$	962(10)	2160(14)	1021(11)	1510(11)	1551(11)	1055(11)	3879(54)	3660(67)	960(10)
		$\Omega^{(3)}$	1110(11)	1690(13)	1061(11)	2196(14)	1863(13)	1180(13)	3757(68)	3714(86)	1113(13)
	2	$\Omega^{(1)}$	586(10)	1621(20)	611(10)	968(10)	963(10)	595(10)	3710(58)	3742(63)	584(10)
		$\Omega^{(2)}$	949(10)	1954(20)	1081(10)	2042(13)	1758(14)	960(10)	3925(49)	3918(51)	951(11)
		$\Omega^{(3)}$	680(8)	1124(13)	693(8)	1282(11)	1256(13)	705(8)	3471(71)	3264(38)	677(8)
20	1	$\Omega^{(1)}$	1076(7)	1889(10)	1179(8)	1925(13)	1582(10)	1120(8)	3726(36)	3692(39)	1074(8)
		$\Omega^{(2)}$	1155(6)	1949(11)	1258(6)	2208(13)	1810(8)	1203(6)	3607(38)	3779(31)	1152(6)
		$\Omega^{(3)}$	1368(7)	2237(11)	1439(6)	2208(10)	2053(8)	1461(7)	3861(29)	3828(35)	1368(7)
	2	$\Omega^{(1)}$	969(6)	1771(10)	1054(6)	1572(8)	1490(8)	978(6)	3503(19)	3612(40)	970(6)
		$\Omega^{(2)}$	959(6)	1539(8)	1049(4)	1843(7)	1509(7)	958(4)	3685(23)	3707(33)	957(6)
		$\Omega^{(3)}$	1159(8)	1677(7)	1266(8)	1770(8)	1600(7)	1166(8)	3690(19)	3730(26)	1161(8)
40	1	$\Omega^{(1)}$	1198(4)	1884(6)	1289(4)	2218(7)	1840(6)	1228(4)	3904(29)	3892(14)	1196(4)
		$\Omega^{(2)}$	1066(4)	1935(7)	1148(4)	1825(8)	1709(7)	1127(6)	3686(47)	3823(24)	1065(4)
		$\Omega^{(3)}$	1296(4)	2084(8)	1393(4)	2292(7)	2017(6)	1378(4)	3779(20)	3698(31)	1293(4)
	2	$\Omega^{(1)}$	997(6)	1911(6)	1084(6)	1717(10)	1541(8)	1005(6)	3709(19)	3633(19)	1000(6)
		$\Omega^{(2)}$	1165(4)	1748(6)	1286(4)	1782(4)	1584(4)	1171(4)	3806(51)	3849(69)	1165(4)
		$\Omega^{(3)}$	1029(7)	1817(8)	1073(7)	2025(13)	1558(10)	1047(7)	3624(31)	3618(48)	1028(7)

TABLE II
TEST ERRORS (STANDARD ERRORS IN PARENTHESES) IN ORDER OF 10^{-4} OF VARIOUS METHODS UNDER THE HAMMING LOSS OVER 100 SIMULATIONS. TYPES 1 AND 2 DENOTE NON-SEPARABLE AND SEPARABLE CASES AND $\Omega^{(1)}$–$\Omega^{(3)}$ REPRESENT THREE TEMPORAL DEPENDENCIES.

K	Type	Ω	Our	LSVM-BR	NSVM-BR	LSVM	NSVM	RF	LSTM-FCN	ALSTM-FCN	Oracle
5	1	$\Omega^{(1)}$	1186(14)	2040(27)	115(13)3	2193(18)	1926(16)	1222(14)	4266(77)	4139(82)	1184(14)
		$\Omega^{(2)}$	1231(12)	2634(17)	1287(14)	1906(15)	1969(15)	1347(14)	4812(69)	4579(80)	1229(13)
		$\Omega^{(3)}$	1416(17)	2072(16)	1327(14)	2777(19)	2357(16)	1510(15)	4697(83)	4687(103)	1420(15)
	2	$\Omega^{(1)}$	756(12)	2011(19)	775(12)	1210(13)	1202(13)	767(12)	4710(77)	4753(84)	753(12)
		$\Omega^{(2)}$	1223(14)	2397(23)	1363(13)	2547(17)	2189(18)	1236(13)	.4809(65)	4797(67)	1226(14)
		$\Omega^{(3)}$	858(11)	1440(16)	865(11)	1637(14)	1607(15)	890(11)	4489(87)	4269(46)	855(11)
20	1	$\Omega^{(1)}$	1346(9)	2326(12)	1469(11)	2367(16)	1968(13)	1402(1)	4618(48)	4584(48)	1345(1)
		$\Omega^{(2)}$	1449(7)	2450(14)	1576(7)	2750(16)	2268(11)	1510(7)	4506(47)	4721(38)	1446(7)
		$\Omega^{(3)}$	1718(8)	2766(13)	1796(7)	2773(12)	2575(10)	1837(8)	4793(33)	4755(42)	1719(9)
	2	$\Omega^{(1)}$	1211(7)	2230(12)	1315(7)	1968(1)	1861(1)	1222(7)	4404(25)	4535(49	1213(6)
		$\Omega^{(2)}$	1194(7)	1936(10)	1303(6)	2306(8)	1883(8)	1195(6)	4593(29)	4618(40)	1192(7)
		$\Omega^{(3)}$	1448(11)	2107(10)	1578(10)	2212(10)	1998(9)	1458(10)	4635(22)	4686(33)	1451(10)
40	1	$\Omega^{(1)}$	1493(5)	2361(7)	1604(5)	2768(9)	2293(6)	1532(6)	4885(36)	4870(18)	1491(5)
		$\Omega^{(2)}$	1335(6)	2412(9)	1431(6)	2285(10)	2137(9)	1413(6)	4592(57)	4758(27)	1333(6)
		$\Omega^{(3)}$	1616(5)	2611(11)	1735(5)	2862(10)	2516(7)	1718(5)	4737(26)	4631(36)	1613(6)
	2	$\Omega^{(1)}$	1241(7)	2410(8)	1347(7)	2145(12)	1920(11)	1250(7)	4650(24)	4558(23)	1244(7)
		$\Omega^{(2)}$	1455(6)	2193(8)	1604(6)	2234(6)	1982(6)	1463(5)	4767(63)	4816(86)	1455(6)
		$\Omega^{(3)}$	1292(8)	2283(10)	1344(9)	2541(16)	1959(13)	1314(8)	4513(36)	4523(60)	1291(8)

account in the model.

Concerning computational speed, as indicated in Table III, SVM based methods run faster, including LSVM-BR, NSVM-BR, LSVM, and NSVM, due to the efficient software–*LIBLINEAR [36] and LIBSVM [37]*. Moreover, they run at least 50% faster than the methods based on random forests, including Our, RF and Oracle. Generally, the methods accounting for the temporal dependence run faster than those ignoring it. For example, Our method runs as fast as the oracle method, while improves the speed of the weighted random forests (RF) by about 30%.

TABLE III
AVERAGE RUNTIME IN SECONDS OF VARIOUS METHODS IN TABLE I FOR ONE 100 SIMULATION. HERE $\Omega^{(1)}$–$\Omega^{(3)}$ REPRESENT THREE TEMPORAL DEPENDENCIES.

K	type	Ω	Our	LSVM-BR	NSVM-BR	LSVM	NSVM	RF	LSTM-FCN	ALSTM-FCN	Oracle
5	1	$\Omega^{(1)}$	21.67	0.45	3.63	0.28	10.33	33.24	398.65	425.85	21.53
		$\Omega^{(2)}$	21.28	0.55	3.21	0.28	11.62	34.58	408.21	449.63	21.32
		$\Omega^{(3)}$	25.00	0.52	4.26	0.30	13.40	38.69	411.38	463.24	24.85
	2	$\Omega^{(1)}$	14.97	0.64	2.06	0.25	8.80	23.04	385.12	441.96	14.79
		$\Omega^{(2)}$	21.93	0.83	3.54	0.27	12.10	32.61	399.65	462.67	21.89
		$\Omega^{(3)}$	16.94	0.69	2.56	0.26	9.75	27.18	392.81	459.29	16.83
20	1	$\Omega^{(1)}$	93.20	1.20	12.51	0.73	34.53	138.82	1625.25	1844.85	93.20
		$\Omega^{(2)}$	90.35	1.68	13.40	0.86	43.69	141.93	1663.84	1869.57	90.30
		$\Omega^{(3)}$	96.23	2.10	13.45	0.84	45.05	154.74	1695.79	1886.63	96.19
	2	$\Omega^{(1)}$	78.21	3.24	12.50	0.95	44.80	116.90	1612.62	1821.74	78.04
		$\Omega^{(2)}$	88.73	2.09	16.31	0.99	46.26	125.20	1698.65	1877.38	88.77
		$\Omega^{(3)}$	109.16	1.81	22.83	1.05	46.05	142.22	1672.93	1854.69	109.08
40	1	$\Omega^{(1)}$	185.30	2.66	23.86	1.52	79.43	286.95	3225.32	3589.16	185.13
		$\Omega^{(2)}$	167.47	3.00	21.83	1.85	96.32	274.87	3232.25	3528.76	168.70
		$\Omega^{(3)}$	198.82	4.33	30.55	2.15	109.72	312.87	3276.67	3716.52	198.48
	2	$\Omega^{(1)}$	158.29	6.42	23.15	1.78	89.80	242.19	3255.48	3632.27	158.12
		$\Omega^{(2)}$	218.38	3.75	45.67	1.81	92.90	288.74	3305.16	3689.61	218.47
		$\Omega^{(3)}$	177.18	4.28	28.67	1.81	91.92	267.09	3298.57	3710.54	176.72

B. Video data analysis

This section applies the proposed method to a benchmark dataset in *ActivityNet Captions dataset* [38] (https://cs.stanford.edu/people/ranjaykrishna/densevid/). This dataset is comprised of annotated videos with captions giving a short sentence description as well as their duration, where each sentence is associated with one segment of the video, depicting multiple distinct events. Note that multiple events may occur simultaneously over a long or short period of time. The reader may consult http://activity-net.org/challenges/2017/captioning.html for more details.

1) Preprocessing: Given a video $V = (v_t)$, each caption is expressed as $\{(t_1, t_2), caption\}$, where the caption describes events occurred in a time interval (t_1, t_2). To create labels for classification, we employ the Stanford NLP parser to extract subjects in addition to the corresponding verbs from main entitles, treating same verbs in different tenses as one category. More specifically, a parser yields a single word with a part-of-sentence tags as well as the grammar between words for a sentence. For example, a parsed sentence *('holds', 'VBZ'), 'dobj', ('arms', 'NNS')* indicates that *'arms'* is a direct object of the verb "*holds*", and the main action in this sentence is "*hold arms*". Then we code a main action as 1/-1 in the presence/absence in a video. For an entire video, we rank actions in parse sentences in captions by the number of occurrences in the video, and choose the top 15 actions to generate labels. As a result, the label $\boldsymbol{Y}_t$ is a 15-dimensional vector, one for each action, with $Y_t^k = \pm 1$ indicating that the k-th action is present/absent at time t.

To generate latent states for our method, we calculate the sentence embeddings via *sentence2vec* [39]. For any two sentences j and k, a similarity measure is defined as $\exp(\frac{\boldsymbol{u}_j^T \boldsymbol{u}_k}{||\boldsymbol{u}_j||_2 ||\boldsymbol{u}_k||_2})$ based on embeddings, where $\boldsymbol{u}_j$ is the sentence vector representation of sentence j. Then we perform clustering analysis based on the similarity measure, resulting that similar sentences are within one cluster serving as latent states. Therefore, two latent states are constructed, coded as Z_i; $i = 1, 2$.

For feature generation, we apply a pre-trained C3D deep network [40] to *ActivityNet*, and sample one for every 16 frames of a video, resulting 500-dimensional feature vectors at n sampled time points, or a $n \times 500$ feature matrix, where n is the number of sampled frames.

Preprocessing yields a sample $\boldsymbol{U}_t = \{\boldsymbol{X}_t, \boldsymbol{Y}_t, Z_t; t = 1, \cdots, T\}$, where at time t, $\boldsymbol{X}_t$ is a 500-dimensional feature vector, $\boldsymbol{Y}_t = (Y_t^1, \cdots, Y_t^{15})$ is a 15-dimensional label vector, and Z_t represents the latent state.

In order to retain the natural temporal dependence of each original video, we reserve the first 60% of a video's frames for initial training and subsequently perform a one-step forward prediction starting from there. In particular, given a video with N frames, the initial training set includes frames from 1 to $0.6N$, while the $(0.6N + 1)$th and $(0.6N + 2)$th frames are used for tuning and testing. Then, we roll one step forward, that is, the training, tuning, and testing sets involve frames 2 to $0.6N + 1$, the $(.6N + 2)$th frame, and the $(0.6N + 3)$th frame, respectively. Finally this process iterates until the last frame is included in the testing set.

Finally, due to the computational constraint in *R-package –mhsmm* [41], we randomly choose 75 dimensions from 500-dimension features in the training, tuning and testing sets. Then this sampling process is repeated for 100 times, and we report the mean and standard deviation of the evaluation metrics based these 100 replications.

2) Numerical results: This section compares the proposed method against two state-of-art deep neural networks, namely long-short term memory fully convolutional network (LSTM-FCN) and the attention long-short term memory fully convolutional network (ALSTM-FCN) [42] in one benchmark example *ActivityNet Captions dataset*. For a reference, also included in the comparison are the aforementioned five competitors LVSM-BR, NSVM-BR, LSVM, NSVM, and RF. For LSTM-FCN and ALSTM-FCN, we use the Keras library with the Tensorflow backend to train. The codes are available at https://github.com/titu1994/LSTM-FCN.

With 100 replications of 500-step forward prediction, we summarize the performances under the weighted loss (2) and the Hamming loss in Table IV, in addition to the runtimes.

As suggested in Table IV, the proposed method and the weighted random forests (RF) are the best performers and perform similarly in terms of the weighted loss, while the proposed method outperforms the weighted random forests in terms of the Hamming distance. Overall, the proposed method outperforms other competitors. In terms of the weighted loss, the amount of improvement of the proposed method is 71.7% over LSVM-BR, 51.6% over NSVM-BR, 60.7% over LSVM, 49.6% over NSVM, 80.3% over LSTM-FCN, and 87.1% over ALSTM-FCN. In terms of the Hamming loss, the amount of improvement of the proposed method is 13.6% over LSVM-BR, 4.4% over RF, 18.9% over LSTM-FCN, and 43.7% over ALSTM-FCN. The improvement is primarily attributed to that the proposed method accounts for label and temporal dependencies, while other methods have not modeled neither dependence or both dependencies. This result is consistent with the simulation results. In terms of the runtime, the proposed method and the weighted random forests are among the slowest, followed by LSTM-FCN and ALSTM-FCN.

In summary, the proposed method performs best in terms of the weighted loss, while it runs faster than its competitors that deliver similar performances.

TABLE IV
TEST ERRORS (STANDARD ERRORS IN PARENTHESES) UNDER THE WEIGHTED AND HAMMING LOSSES AND AVERAGE RUNTIME IN SECONDS OF VARIOUS METHODS OVER 100 RANDOM SAMPLES FROM THE VIDEO SEQUENCE.

	Weighted Loss	Hamming Loss	Runtime(seconds)
Our	.0060(.00003)	.0389(.0002)	73.13
LSVM-BR	.0212(.00042)	.0450(.0005)	1.47
NSVM-BR	.0124(.00013)	.0279(.0002)	11.66
LSVM	.0153(.00016)	.0316(.0002)	1.23
NSVM	.0119(.00013)	.0274(.0002)	19.18
RF	.0061(.00003)	.0407(.0002)	103.52
LSTM-FCN [42]	.0306(.00382)	.0480(.004)	2828.15
ALSTM-FCN [42]	.0466(.00927)	.0691(.010)	3143.40

V. DISCUSSION

This article introduces a new framework of nonlinear multilabel classification for time-series predictors under a weighted loss separating false positives from negatives. The framework uses a hidden Markov model to describe local piecewise constancy of the time series, motivated primarily from the application of video categorization, where each transition state of the hidden Markov model corresponds to one scene of a video. Moreover,

it has accounted for both temporal dependency as well as label dependency, particularly semantic label dependency, simultaneously. Computationally, we design a scalable algorithm solving the corresponding nonconvex minimization. Numerically, we demonstrate the utility of the proposed method by comparing with its competitors ignoring either the temporal structure or label dependency in simulated and real benchmark examples. Theoretically, we establish consistency of the proposed classifier, which guarantees the recovery of the optimal performance of the Bayes classifier. While the proposed method is applicable to other situations as well, further investigation is necessary.

REFERENCES

[1] A. Elisseeff and J. Weston, "A kernel method for multi-labelled classification," in *Advances in neural information processing systems*, 2002, pp. 681–687.

[2] T. Li, M. Ogihara, and Q. Li, "A comparative study on content-based music genre classification," in *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*. ACM, 2003, pp. 282–289.

[3] M. R. Boutell, J. Luo, X. Shen, and C. M. Brown, "Learning multi-label scene classification," *Pattern recognition*, vol. 37, no. 9, pp. 1757–1771, 2004.

[4] G. Tsoumakas and I. Katakis, "Multi-label classification: An overview," *International Journal of Data Warehousing and Mining (IJDWM)*, vol. 3, no. 3, pp. 1–13, 2007.

[5] D. R. Cox, "The analysis of multivariate binary data," *Applied statistics*, pp. 113–120, 1972.

[6] V. Carey, S. L. Zeger, and P. Diggle, "Modelling multivariate binary data with alternating logistic regressions," *Biometrika*, vol. 80, no. 3, pp. 517–526, 1993.

[7] J. F. Pendergast, S. J. Gange, M. A. Newton, M. J. Lindstrom, M. Palta, and M. R. Fisher, "A survey of methods for analyzing clustered binary response data," *International Statistical Review/Revue Internationale de Statistique*, pp. 89–118, 1996.

[8] L. Breiman and J. Friedman, "Predicting multivariate responses in multiple linear regression," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 59, pp. 3–54, 1997.

[9] J. Read, P. Reutemann, B. Pfahringer, and G. Holmes, "Meka: a multi-label/multi-target extension to weka," *The Journal of Machine Learning Research*, vol. 17, no. 1, pp. 667–671, 2016.

[10] J. Guo, E. Levina, G. Michailidis, and J. Zhu, "Joint estimation of multiple graphical models," *Biometrika*, vol. 98, no. 1, p. 1, 2011.

[11] J. Cheng, E. Levina, P. Wang, and J. Zhu, "Sparse ising model with covariates," *Biometrics*, vol. 70, no. 4, pp. 943–953, 2014.

[12] I. Tsochantaridis, T. Hofmann, T. Joachims, and Y. Altun, "Support vector machine learning for interdependent and structured output spaces," in *Proceedings of the twenty-first international conference on machine learning*. ACM, 2004, p. 104.

[13] V. Gjorgjioski, D. Kocev, and S. Džeroski, "Comparison of distances for multi-label classification with pcts," in *Proceedings of the Slovenian KDD Conference on Data Mining and Data Warehouses (SiKDD'11)*, 2011.

[14] Y. Zhu, X. Shen, H. Jiang, and W. Wong, "Collaborative multilabel classification," *Submitted*, 2019.

[15] L. Wang, Y. Qiao, and X. Tang, "Action recognition and detection by combining motion and appearance features," in *THUMOS Action Recognition challenge*, 2014, pp. 1–6.

[16] L. Wang, Y. Xiong, Z. Wang, Y. Qiao, D. Lin, X. Tang, and L. V. Gool, "Temporal segment networks: Towards good practices for deep action recognition," in *ECCV (8)*, ser. Lecture Notes in Computer Science, vol. 9912. Springer, 2016, pp. 20–36.

[17] F. Karim, S. Majumdar, H. Darabi, and S. Harford, "Multivariate lstm-fcns for time series classification," *Neural Networks*, 01 2018.

[18] C. Orsenigo and C. Vercellis, "Combining discrete svm and fixed cardinality warping distances for multivariate time series classification," *Pattern Recognition*, vol. 43, pp. 3787–3794, 11 2010.

[19] S. Seto, W. Zhang, and Y. Zhou, "Multivariate time series classification using dynamic time warping template selection for human activity recognition," *2015 IEEE Symposium Series on Computational Intelligence*, pp. 1399–1406, 2015.

[20] W. Pei, H. Dibeklioglu, D. M. J. Tax, and L. van der Maaten, "Multivariate time-series classification using the hidden-unit logistic model," *IEEE Trans. Neural Netw. Learning Syst.*, vol. 29, no. 4, pp. 920–931, 2018.

[21] T. Jaakkola, M. Diekhans, and D. Haussler, "A discriminative framework for detecting remote protein homologies," 1999.

[22] L. van der Maaten, "Learning discriminative fisher kernels," in *ICML*. Omnipress, 2011, pp. 217–224.

[23] M. G. Baydogan and G. Runger, "Time series representation and similarity based on local autopatterns," *Data Min. Knowl. Discov.*, vol. 30, no. 2, pp. 476–509, Mar. 2016. [Online]. Available: http://dx.doi.org/10.1007/s10618-015-0425-y

[24] P. Schäfer and U. Leser, "Multivariate time series classification with WEASEL+MUSE," *CoRR*, vol. abs/1711.11343, 2017.

[25] K. S. Tuncel and M. G. Baydogan, "Autoregressive forests for multivariate time series modeling," *Pattern Recognition*, vol. 73, pp. 202–215, 2018.

[26] P. Tseng, "Convergence of a block coordinate descent method for nondifferentiable minimization," *Journal of optimization theory and applications*, vol. 109, no. 3, pp. 475–494, 2001.

[27] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[28] J. Friedman, T. Hastie, and R. Tibshirani, "Sparse inverse covariance estimation with the graphical lasso," *Biostatistics*, vol. 9, no. 3, pp. 432–441, 2008.

[29] G. D. Forney, "The viterbi algorithm," *Proceedings of the IEEE*, vol. 61, no. 3, pp. 268–278, 1973.
[30] M. Wright and A. Ziegler, "ranger: A fast implementation of random forests for high dimensional data in c++ and r," *Journal of Statistical Software, Articles*, vol. 77, no. 1, pp. 1–17, 2017. [Online]. Available: https://www.jstatsoft.org/v077/i01
[31] C.-J. Hsieh, I. S. Dhillon, P. K. Ravikumar, and M. A. Sustik, "Sparse inverse covariance matrix estimation using quadratic approximation," in *Advances in neural information processing systems*, 2011, pp. 2330–2338.
[32] G. Biau, "Analysis of a random forests model," *Journal of Machine Learning Research*, vol. 13, no. Apr, pp. 1063–1095, 2012.
[33] G. Biau, L. Devroye, and G. Lugosi, "Consistency of random forests and other averaging classifiers," *Journal of Machine Learning Research*, vol. 9, no. Sep, pp. 2015–2033, 2008.
[34] J. Tadjuidje Kamgaing, "Maximum likelihood estimators for multivariate hidden markov mixture models," Tech. Rep., 2013.
[35] A. J. Rothman, E. Levina, and J. Zhu, "A new approach to cholesky-based covariance regularization in high dimensions," *Biometrika*, vol. 97, no. 3, pp. 539–550, 2010.
[36] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin, "Liblinear: A library for large linear classification," *J. Mach. Learn. Res.*, vol. 9, pp. 1871–1874, Jun. 2008. [Online]. Available: http://dl.acm.org/citation.cfm?id=1390681.1442794
[37] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, pp. 27:1–27:27, 2011, software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.
[38] R. Krishna, K. Hata, F. Ren, L. Fei-Fei, and J. C. Niebles, "Dense-captioning events in videos," in *International Conference on Computer Vision (ICCV)*, 2017.
[39] Q. Le and T. Mikolov, "Distributed representations of sentences and documents," in *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32*, ser. ICML'14. JMLR.org, 2014, pp. II–1188–II–1196. [Online]. Available: http://dl.acm.org/citation.cfm?id=3044805.3045025
[40] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning spatiotemporal features with 3d convolutional networks," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 4489–4497.
[41] J. O'Connell and S. Højsgaard, "Hidden semi markov models for multiple observation sequences: The mhsmm package for r," *Journal of Statistical Software, Articles*, vol. 39, no. 4, pp. 1–22, 2011. [Online]. Available: https://www.jstatsoft.org/v039/i04
[42] F. Karim, S. Majumdar, H. Darabi, and S. Chen, "Lstm fully convolutional networks for time series classification," *IEEE Access*, vol. 6, pp. 1662–1669, 2018.