Time-sensitive Classification of Behavioral Data

Shin Ando

Einoshin Suzuki

Abstract

In this paper, we address a classification task under a time-sensitive setting, in which the amount of observation required to make a prediction is viewed as a practical cost. Such a setting is intrinsic in many systems where the potential reward of the action against the predicted event depends on the response time, e.g., surveillance/warning and diagnostic applications. Meanwhile, predictions are usually less reliable when based on fewer observations, i.e., there exists a trade-off between such temporal cost and the accuracy. We address the task as a classification of subsequences in a time series. The goal is to predict the occurrences of events from subsequent observations and to learn when to commit to the prediction considering the trade-off. We propose an ensemble of classifiers which respectively makes predictions based on subsequences of different lengths. The prediction of the ensemble is given by the earliest confident prediction among the individual classifiers. We propose a cutting-plane algorithm for jointly training an ensemble of linear classifiers considering their temporal dependence. We compare the proposed algorithm against conventional approaches over a collection of behavioral trajectory data.

keywords Time-sensitive Classification, Subsequence Classification, Behavior Analysis

1 Introduction

Decision making in a practical system is often time-sensitive, i.e., there are incentives to make a decision earlier rather than later. For example, in surveillance and warning systems, earlier detection of dangerous behaviors or hazardous events allows for better response time and possibly reduced damage. For a medical diagnostic system, predicting the pathology of patients with fewer test observations may allow for better or less expensive options of treatment. On the other hand, there is generally a trade-off between the earliness and the accuracy of predictions, i.e., those made without sufficient observations are naturally unreliable. In this paper, we refer to the problem of such a setting as Time-sensitive Classification (TSC). The goal of TSC is learning to predict the target event from ensuing observations and also when to commit to a response considering the accuracy and the potential rewards or costs of different options.

The issue of varying costs in classification is important in practical applications and has been addressed extensively in Cost-sensitive Learning (CSL) [1]. In cost-sensitive classification, the goal is to learn a classifier that minimizes the total cost, given the class-dependent costs of misclassification. However, extending the CSL framework to the time-sensitive case is not straightforward, as it involves decisions to reserve or to commit to the prediction on each instance.

The task of classifying time series by its earlier observations has been addressed previously in Early Classification of Time Series (ECTS) [2, 3, 4]. Its goal is to classify a time series instance based on their prefix, i.e., initial subsequences, under the condition that a certain level of accuracy is maintained. The implementation of ECTS, however, only incorporates the desired accuracy as the problem-dependent knowledge and therefore cannot account for the time-varying costs.

In the TSC problem setting, the predictions are made based on the latest series of observations. This setting is naturally formalized as a classification of the subsequence of time series (STS). The classifier is trained to predict the class value of an event at the start of each subsequence in a given time series. To provide grounds for an action/inaction and avoid premature errors, each prediction should also be accompanied by a confidence level. As the classifier makes predictions successively using the most recent observations, one can commit to a response when the level of confidence or the expected loss becomes acceptable.

In order to classify STS data of successively increasing lengths, we construct an ensemble of classifiers each taking a subsequence of a different length as an input. Those taking the shorter subsequence can make an earlier prediction, but if its confidence is low, the response can be reserved until more observations become available. The prediction of the ensemble is given by the earliest confident prediction. We refer to this scheme as
Time-sensitive Ensemble (TSE).

The individual classifiers which compose TSE have strong dependences among one another. For example, depending on the prediction of an earlier classifier, the other classifiers may or may not need to predict the same event instance. In turn, the performances of later classifiers affect whether to be cautious or be bold with earlier predictions. The preferable composition may depend on the domain knowledge such as the cost of errors at different timings.

In this paper, we introduce the knowledge of the preference or the varying cost of errors as a set of weights in the definition of the loss functions. The training of the ensemble is formulated using the joint loss among the classifiers. Due to the time-dependent aspect of TSE, its training also entails the problem of choosing the timing of prediction, i.e., assigning a subset of training instances to each classifier. The main contribution of this work is formalizing two problems: finding the assignments and training the classifiers with the assigned data, based on a joint loss function. The problem is addressed by a combination of an extended cutting-plane algorithm and steepest descent. We present an iterative algorithm which addresses two problems in alternating steps and converges to the local minimum of a regularized loss function.

We conduct an empirical study of the proposed algorithm with a focus on the task of behavior analysis. We evaluate the performance of the proposed algorithm using the trajectories from multi-agent task experiments and human activities. The rest of this paper is organized as follows. Section 2 describes the related work, and Section 3 reviews the basic formulation of the cutting-plane algorithm. Sections 4 and 5 describe our original contributions. The former introduces the formulations of the Time-sensitive Classification problem and the training of Time-sensitive Ensemble. The latter describes the cutting-plane algorithm for training the ensemble. In Sections 6 and 7, we present the empirical study and our conclusion, respectively.

2 Related Work

2.1 Time Series Classification Temporal and sequential data is an important subject for data mining in many domains, e.g., financial, medical, and genome informatics applications [2, 6]. Among the existing methods, the k-nearest neighbor, SVM, and the model-based classifiers are shown to be effective for classifying sequential data [2]. In a time-sensitive classification, the classification model needs to make predictions on subsequences of different lengths and also determine the timing of the prediction. Time-sensitive classification is a new, challenging problem for the standard classification models that take input from a fixed dimension. In the following sections, we discuss some of the related problems in machine learning.

2.2 Early Classification of Time Series Early Classification of Time Series (ECTS) [3] is an interesting application of sequence classification. Its goal is described as a) affirming the earliest time of reliable classification and b) retaining a level of accuracy specified by the user. In [4], ECTS is implemented as a hybrid instance-based learning combining 1-NN classification and hierarchical clustering in a sophisticated manner. Minimum Prediction Length (MPL) is introduced as the length to which each training instance can be cut short, while maintaining the structure of single linkage clustering at the full length. Each training instance can be used to predict test set instances to which it is the nearest neighbor, based on the initial MPL points. Additionally, ECTS incorporates a concept of serial to practice caution.

While previous studies on ECTS have achieved their initial goals, there remain several issues for practical applications. First, it is difficult to integrate application knowledge such as the cost variation, since the training of ECTS is based solely on the constraint of accuracy. For example, it is difficult to adjust the ECTS classifier to make a bold early prediction even when the trade-off is favorable. Secondly, the sensitivity of the single linkage clustering to noisy data may be problematic for real-world applications. Finally, when target events occur in a large time-scale, computing the clustering from the full length to the MPL incrementally is a significant computational burden.

In TSC, the first issue is addressed by introducing a set of weights to reflect the varying cost in the loss function. The second issue is addressed by employing a feature-based approach, which generally achieves better generalization compared to the instance-based approach. The third issue is addressed in TSE by allowing the user to designate the timing, at which the potential reward of the decision would change.

2.3 Cost-sensitive Classification Cost-sensitive learning (CSL) [1] is motivated by real-world decision making tasks where different decisions and results can yield varying costs. There are two categories of CSL methods: the embedded and the meta-learning approaches. The former attempts to learn a classifier that minimizes the expected cost directly, while the latter constructs a cost-sensitive classifier based on the outputs of existing algorithms.

The key difference of TSC from CSL is the time-dependent variation of the cost. The CSL classifiers
aimed at minimizing the expected cost based on the probability of each class, such as MetaCost [7], does not extend naturally to the time-sensitive setting as the class probabilities and the expected cost when the prediction is reserved for later are unknown. Further, as many meta-learning CSL methods are computationally intensive, it is difficult to learn a collection of classifiers required to make predictions at different timings. For example, Empirical Thresholding (ET) [14] makes meta-predictions based on the scores of an existing classifier. Given the scores over the validation data from a trained classifier, ET selects a threshold that achieves the least total cost by an exhaustive comparison over all feasible values. In a TSC setting, ET must choose a combination of thresholds for multiple classifiers, which requires an exponential number of comparisons.

3 Preliminaries

This section review the basic formulation of the cutting plane algorithm for solving a quadratic programming problem, which provides the basis for our work described in Section 4. In [5], a cutting-plane algorithm for training a linear SVM is proposed. Cutting-plane (CP) method [8] is a technique employed in various constrained optimization problems, such as linear programming and relaxed integer programming. Commonly the CP algorithm solves an optimization problem iteratively subject to a working set of constraints. In each iteration, a constraint separating the tentative solution from the desired solutions, or a cut, is incremented to the working set. Finding the cut is the core problem of the cutting-plane method that is specific to the formulation of the optimization.

Let \( \mathcal{X} = \{x_i\}_{i=1}^n \) denote the observed data and \( y_i \in \{-1, 1\} \) the true class label of \( x_i \). Let \( f : x \in \mathbb{R}^l \rightarrow \mathbb{R} \) denote the discriminative function of a linear SVM, where \( \lambda \) denotes the dimension of the input domain. \( f \) is parametrized by a weight vector \( w \), such that \( f(x_i) = w^\top x_i \). For brevity, the constant parameter \( b \) is included in \( w \), i.e., \( w = (b, w_1, \ldots, w_\lambda) \) and \( x_i = (1, x_{i1}, \ldots, x_{i\lambda}) \).

Training SVM to predict \( y_i \) from the given data is a quadratic programming problem. The formulation in one slack variable \( \xi \) proposed in [5].

\[
\text{arg min}_{w, \xi \geq 0} \frac{1}{2} \|w\|^2 + C \xi \\
\text{subject to } \forall z \in \{0, 1\}^\eta:
\]

\[
\frac{1}{\eta} \sum_{i=1}^\eta z_i y_i (w^\top x_i) \geq \frac{1}{\eta} \sum_{i=1}^\eta z_i - \xi
\]

where \( \mathcal{Z} \subset \{0, 1\}^\eta \) is a set of index vectors. Each \( z \) represents a subset of \( \mathcal{X} \) and corresponds to a constraint through (3.2). The cutting-plane algorithm iterates solving (3.1) subject to all \( z \) in a working set \( \mathcal{Z} \) until it reaches the optimum.

In this paper, we extend the cutting-plane formulation in order to train an ensemble of classifiers jointly considering the dependence among them.

4 Problem Formulation

4.1 Time-sensitive Classification

Let \( \mathcal{X} = \{x_i\}_{i=1}^\tau \) denote the observed time series and \( \mathcal{Y} = \{y_i\}_{i=1}^\tau \) the class labels. The value of \( y_i \in \{1, -1\} \) represents the class of the event occurring at time \( t \). The goal of TSC is to predict \( y_t \) early and accurately based on the subsequent observations \( (x_t, x_{t+1}, \ldots) \).

This formulation accommodates the characteristics of the behavioral data, which is the primary subject in our empirical study. The intent or the purpose that induces the behavior is usually determined but not observable at the start, and it becomes apparent only with the progression of the behavior.

Let \( v_t(t) = (x_t, \ldots, x_{t+\ell-1}) \) denote a subsequence of \( \mathcal{X} \) and \( y'_t(t) \) the prediction of \( y_t \) after those \( \ell \) observations. \( y'_t(t) \) is given by a discriminative function \( f_{\ell} : v \in \mathbb{R}^\ell \rightarrow \mathbb{R} \) and a threshold \( \theta_{\ell} \),

\[
y'_{\ell}(t) = \begin{cases} 
1 & \text{if } f_{\ell}(v_{\ell}(t)) \geq \theta_{\ell} \\
-1 & \text{if } f_{\ell}(v_{\ell}(t)) \leq -\theta_{\ell} \\
0 & \text{otherwise}
\end{cases}
\]

The values \( y'_{\ell}(t) = \pm 1 \) indicate the commitment to a responsive action, based on the confidence of the prediction. On the other hand, if \( y'_{\ell}(t) = 0 \), the decision would be reserved until more observation becomes available. \( \theta_{\ell} \) scales the interval between the opposing classes.

In practice, there usually exists some deadline, or limited window of time, for any response to be effective. We assume such decision windows are given for a preset of actions and denote it by \( \mathcal{K} = \{\lambda_k\}_{k=1}^n \). Since \( x_t \) is observed sequentially, the observations available at the respective decision windows is given by \( v_{\lambda}(t) : \lambda \in \mathcal{K} \).

Given \( \mathcal{K} \), we learn a set of functions \( \mathcal{F} = \{f_{\lambda}\}_{\lambda \in \mathcal{K}} \). Each function makes a prediction from \( \lambda_k \) observations by (4.3). From \( \mathcal{F} \), we obtain a series of predictions until either a decision is made or no decision is made in time,

\[
y(t) = [y'_{\lambda_1}(t), y'_{\lambda_2}(t), \ldots, y'_{\lambda_p}(t)]
\]

where \( \lambda_p \) denotes the length of time between the occurrence of an event at \( t \) and the first confident prediction on the class of the event. We refer to this length of time as the decision length and denote by \( \Lambda(t) \).

The collective prediction of the ensemble \( y(t) \in \)
\{ -1, 0, 1 \} is given by
\[ y'(t) = \sum_{\lambda_k : \lambda_k \leq A(t)} y'_{\lambda_k}(t) \]
No decision is made if \( y'(t) = 0 \), i.e., \( \forall \lambda \in K : y_{\lambda}(t) = 0 \).

Given an input \( X \), the final predictions of a Time-sensitive Ensemble \( E(F, K) \) is \( y' = \{ y'(t) \} \). In the case of Fig. 4, the TSE composed of \( f_1 \) and \( f_2 \) yields decisions for the instances in the meshed and shaded regions but makes no decision for those in the central white region.

4.2 Training an SVM Ensemble
We now formulate the training of an ensemble of linear SVMs for TSC. As we have \( k \) decision windows, we use \( v_{\lambda_k}(t) \) instead of \( v_{\lambda_k} \) for a subsequence of \( k \) points, \((x_t, \ldots, x_{t+k-1})\) for brevity. Let \( \mathbf{V}_k = \{ v_{\lambda_k}(t) \}_{t=1}^g \) denote all subsequences of length \( k \) extracted from \( X \). We also use \( v_t \) in place of \( \mathbf{V}_k(t) \), when \( k \) is clear from the context.

Let \( \mathbf{w}_k \) denote the weight vector of the linear SVM taking a \( \lambda_k \)-dimensional vector as its input. It parametrizes the discriminative function \( f_k \)
\[ f_k(v_t) = \mathbf{w}_k \cdot v_t \]

Constructing TSE for a given \( K = \{ \lambda_k \} \) requires training the sets of weights \( W = \{ \mathbf{w}_k \}_{k=1}^g \) and thresholds \( \Theta = \{ \theta_k \}_{k=1}^g \). As discussed in Section 1, our motivation is to do considering the domain knowledge and the temporal dependence of the classifiers.

In the following formulation, a user-defined set of weights \( R = \{ \rho_k \} \) is introduced to represent the time-sensitive aspect of the problem.

**Problem 1.**
\[
\begin{align}
\text{arg min}_{\mathbf{w}_k \geq 0} & \sum_{k=1}^{g} \| \mathbf{w}_k \|^2 + C \xi \\
\text{subject to} & \forall t \in Z \in \mathbf{Z} : \\
\frac{1}{\eta} \sum_{t,k} z_{t,k} y_k(\mathbf{w}_k \cdot v_k(t)) & \geq \frac{1}{\eta} \sum_{t,k} \theta_k z_{t,k} - \xi \\
\end{align}
\]
where \( Z = \{ z_{t,k} \}_{t \times \lambda} \) is an index matrix, representing the assignment of the subsets of \( Y \) to respective classifiers. \( z_{t,k} \) is 1 if the confident prediction of the event at \( t \) comes from \( f_k \), i.e., \( y'(t) = y'_{\lambda_k}(t) \), and 0 otherwise. Since the ensemble commits to the earliest confident prediction, a feasible assignment should satisfy \( \forall j < k : z_{t,j} = 0 \) given \( z_{t,k} = 1 \). \( Z \) denotes the set of all feasible assignments.

In (4.6), the inequality constrains the sum of average margins by the sum of maximum training errors, accepting the violation up to \( \xi \). The error of each classifier is re-scaled by the corresponding threshold in \( \Theta \), which is fixed within the scope of Problem 1, and the summation is taken over the assigned subsets.

Problem 1 minimizes \( \xi \), which is the upper-bound of the sum of errors by respective classifiers, regularized by the weighted sum of their margins over \( Z \). Each \( \rho_k \) is a positive weight, which affects the tightness of the margin of the corresponding SVM. The penalty on the errors made at the \( k \)th decision window is emphasized by a larger \( \rho_k \). Given the domain knowledge, \( R \) can be set to reflect the severity of the error at respective windows. In this study, we consider \( \rho_k \) which monotonically decreases against the increasing \( \lambda_k \) to discourage errors from premature predictions.

To solve Problem 1 by an iterative cutting-plane method, it is necessary to find a cut for any solution of \( W \), i.e., a constraint that it violates but the desired solution does not. Finding those with larger violations among such constraints leads to a faster convergence.

Given a tentative solution \( W^* \), the violation of the constraint (4.6) is given by \( \max \{ 0, V(Z, \Theta) \} \), where
\[ V(Z, \Theta) = \frac{1}{\eta} \sum_{t,k} z_{t,k} \{ \theta_k - y_k(\mathbf{w}_k \cdot v_t) \} - \xi^* \]

We find the cut by maximizing (4.7) with regards to a feasible \( Z \) and \( \Theta \). The problem is formulated as follows.

**Problem 2.**
\[
\begin{align}
\text{arg min}_{\xi_{t,k} \geq 0, \theta_k \geq 0} & \sum_{\theta_k} \theta_k^2 + C \xi \sum_{t,k} \xi_{t,k} \\
\text{subject to} & \forall t \in Z \in \mathbf{Z} : \\
\frac{1}{\eta} \sum_{t,k} z_{t,k} y_k(\mathbf{w}_k \cdot v_k(t)) - \theta_k & \geq -\xi_{t,k} \\
\end{align}
\]

For each \( (t, k) \), a slack variable \( \xi_{t,k} \) represents the violation due to \( v_k(t) \). For any given \( \Theta \), the least sum of \( \xi_{t,k} \) gives the largest violation among all \( Z \in \mathbf{Z} \),
\[ \min \frac{C}{\eta} \sum_{t,k} \xi_{t,k} = \max \frac{C}{\eta} \sum_{t,k} z_{t,k} \{ \theta_k - y_k(\mathbf{w}_k \cdot v_t) \} \]

The optimal solution thus gives the maximal \( V \). Problem 2 can be seen as an optimization of the loss \( V(Z, \Theta) \) regularized by the sum of scaled margins.

For a fixed \( W^* \), \( \xi_{t,k} \) is minimized individually, i.e.,
\[
\begin{align}
\text{arg min}_{\xi_{t,k}} & \sum_{t,k} \xi_{t,k} = \sum_{t,k} \max_{Z} z_{t,k} \{ \theta_k - y_k(\mathbf{w}_k \cdot v_t) \} \\
\end{align}
\]

\( \xi_{t,k} \) thus reduces to 0 if \( v_k(t) \) can be correctly classified. Then by (4.10), \( z_{t,j} \) for all \( j > k \) are constrained to 0. This ensures that the solution \( Z^* \) is a feasible index. With the relaxation of \( z_{t,k} \), (4.8)-(4.10) are converted into a convex optimization problem and solved.
by a steepest descent method. Given a relaxed solution \( Z^* \), the corresponding solution in the integer domain is obtained by replacing all non-zero elements by 1.

Based on Problems 1 and 2, we can implement an iterative algorithm which alternates between minimizing the weighed sum of errors and maximizing the violation of the tentative solution with regards to the subset assignment. Moreover, Problem 1 is solved more conveniently by addressing its subproblems.

**Theorem 4.1.** Given an optimal solution \( \{w_k^*, \xi_k^*\} \) of Problem 3 for each \( k = 1, \ldots, \kappa \),

\[
\mathcal{W}^* = \{w_k^*\}, \xi^* = \sum \xi_k^*
\]

is a solution of Problem 1.

The proof is omitted due to limited space.

Theorem 4.1 shows that for a given \( \Theta \), one may solve Problem 3 independently for each \( k \) instead of Problem 1 and obtain \( \mathcal{W} \) with the same amount of violation. Individually, the formulation of Problem 3 is a quadratic programming problem for training a linear SVM with the minimum margin scaled to \( \theta_k \) [5]. Problem 3 can therefore be solved in parallel for each \( k \) by many existing algorithms.

Note that in Problem 3, even though the loss function can be minimized with regards to \( \theta_k \), it would not necessarily be the global minimum for Problem 1. While solving Problem 1 directly, one is required to fix \( \theta_k \) as \( Z \) changes with \( \theta_k \). We thus attain locally optimal \( \theta_k \) through alternate iteration of Problems 2 and 3 by the algorithm described in the following section.

### Algorithm 1: Cutting-plane Training of TSE

1. Input: Time series \( X = \{x_t\}_{t=1}^T \), labels \( Y = \{y_t\}_{t=1}^T \), decision windows \( K = \{k\}_{k=1}^K \), \( R = \{\rho_k\}_{k=1}^K \)
2. Output: \( \mathcal{W} = \{w_k\}_{k=1}^K \), \( \Theta = \{\theta_k\}_{k=1}^K \)
3. function InitializeZ(): find a \( Z \) feasible for \( \Theta \)
4. Define: \( v_k(t) = (x_t, \ldots, x_{t+\lambda_k-1}) \), slack \( \xi \), \( \{\xi_k\}_{k=1}^K \)
5. Method:
6. Initialize \( Z' \rightarrow \emptyset \), \( \Theta \rightarrow \{\{1\}\}_{k=1}^K \), \( Z \leftarrow \) InitializeZ(\( \Theta \))
7. repeat
8. \( Z' = Z' \cup \{Z\} \)
9. for \( k = 1 \) to \( \kappa \) do
10. \( Z_k = \{z_k\}_{z \in \mathcal{Z}} \)
11. \{\( w_k, \xi_k \)\} \leftarrow \) Solve Problem 3 for \( Z_k \)
12. end for
13. \( \mathcal{W} \leftarrow \{w_k\} \), \( \xi \leftarrow \sum \xi_k \)
14. \{\( \Theta, Z \)\} \leftarrow \) Solve Problem 2 for \( \mathcal{W} \)
15. for \( t = 1, \ldots, \eta, k = 1, \ldots, \kappa \) do
16. \( z_{t,k} \leftarrow \begin{cases} 1 & \text{if } z_{t,k} > 0 \\ 0 & \text{otherwise} \end{cases} \)
17. end for
18. until \( V(\mathcal{W}, \Theta) \) converges or \( Z \in \mathcal{Z}' \)
19. return \( \mathcal{W} \)

In each iteration, we first solve Problem 3 for each \( k \) (Lines 9-12), and then join the solutions to obtain \( \mathcal{W} \) and \( \xi \) (Line 13). We maximize the violation with regards to \( \Theta \) and relaxed \( Z \) (Line 14) and then convert \( Z \) to the integer domain (Lines 15-17). The next iteration continues after adding new \( Z \) to \( \mathcal{Z}' \) (Line 8), unless \( (Z, \Theta) \) does not improve \( V \), or \( Z \) is already found in \( \mathcal{Z}' \) (Line 18).

**Theorem 5.1.** Algorithm 1 improves the solution in each new iteration and converges to a local optimum.

The proof is omitted due to limited space.

### 6 Empirical Evaluation

#### 6.1 Data Descriptions

In our study based on trajectories extracted from human and agent behaviors, analyzing behaviors of target objects in a timely manner is an important task for intelligent surveillance systems. The following sections review the background knowledge of each data. The first two datasets were produced by a JST-ANR research project\(^1\) and have been studied previously in [10, 11]. Two other datasets from the UCI Machine Learning Repository (MLR)\(^2\) are employed. The exact data used in the experiment can be generated from the source code\(^3\).

---

1. [http://www.4.is.titech.ac.jp/~suzuki/jstamr.htm](http://www.4.is.titech.ac.jp/~suzuki/jstamr.htm)
3. [http://an.is.dept.eng.gunma-u.ac.jp/~ando/psd1exp.html](http://an.is.dept.eng.gunma-u.ac.jp/~ando/psd1exp.html)
6.1.1 Anomalous Behavior Trajectories The first dataset is taken from the trials of autonomously controlled robot agents exploring an enclosed testing field [9]. The coordinates of a robot was extracted using edge detection from the sequence of images recorded by a webcam placed above the field. During the experiment, the agent occasionally exhibited an anomalous behavior, which is to pivot repeatedly in the corner of the field. It was unexpected to the designer of the controller program and contradictory to his intent. In [10], the behavior was further analyzed in the context of anomaly detection. For our experiment, we set up TSC to detect the anomalous behavior of the agent early and accurately.

The trajectories of an agent were extracted from three independent trials, respectively referred to as T1, T2, and T3. The class labels of the events were given manually by inspecting each frame of the image sequence. The label +1 is given to those during the pivoting behavior and −1 is given otherwise.

The norm of agent’s velocity computed from each trajectory was employed as the input time series. The set of subsequences is denoted as \( V_\lambda = \{ s_\lambda(t) \}_{t=1}^\eta \). The definition of the velocity and the norm is omitted due to limited space. Fig.1 illustrates the time series of the velocity norm in T1. The \( x \)- and \( y \)-axes represent the time and the value of the norm, respectively. The red bar indicates the occurrence of the pivot behavior.

6.1.2 Interactive Behavior Trajectories The second dataset is taken from a multi-agent task experiment involving interactions of agents. The analysis of interactive behaviors is significantly more complex than that of a single target and presents an important challenge. In the multi-agent task experiment conducted in [12], robot agents equipped with a USB camera attempted to recognize other agents using the real-time images captured by the camera. If an agent recognizes another agent, it attempts to pursue the agent. When the agent loses sight of its target, it abandons the pursuit and explores the enclosed field. We set up the classification task to discriminate the agents in a pursuit and those exploring individually based on the features of their trajectories. As the input time series \( \mathcal{X} \), we employ the cosine distance between the velocities of two agents. The cosine distance between the velocities is a descriptive feature to indicate whether the agents are moving in similar directions. The definition of the cosine distance feature is omitted due to limited space.

We prepared three input time series, each from a pair of trajectories of an independent trial. In each trial, two agents are engaged in a pursuit for the duration of their trajectories. We refer to these trials as P1-3. To prepare examples of baseline behaviors, we employ another three pairs of trajectories of agents which explore the enclosed field individually. We refer to these trajectories as X1-3.

Fig.2 illustrates the time series of cosine distance between trajectories of P1 and X1, shown by green and orange lines, respectively. The \( x \)- and \( y \)-axes represent the time and the cosine distance, respectively.

The subsequences from P1 are labeled with the class value +1 and those of X1 with −1. We refer to the joint set of subsequences from P1 and X1 as PX1. From (P2,X2) and (P3,X3), PX2 and PX3 are prepared in the same manner, respectively.

6.1.3 Sensor Tag Trajectory The third dataset is taken from “Localization Data for Person Activity” in MLR, consisting of 3-D coordinates recorded using sensor tags. Tags were worn on four different parts of a person’s body: both ankles, the belt, and the chest, while performing multiple activities [13]. A key motivation of monitoring the personal behavior is to detect accidents and other potentially dangerous situations from sensor readings. To this end, we set up the classification task to discriminate a sequence of dangerous behaviors from that of normal behaviors with similar sensor readings. We generate one input dataset from the readings of each sensor tag. We refer to those corresponding to the tag on the left and the right ankle, the belt, and the chest as S1, S2, S3, and S4, respectively. The details of the setup are omitted due to limited space.

6.1.4 Handwriting Trajectory The fourth dataset is taken from “Character Trajectories” in MLR, which contains 2858 handwriting trajectories for single pen-
down characters. The trajectories of the pen-tip were captured at 200Hz using a tablet PC. For each timestamp, $x - y$ coordinates and the pen-tip force were recorded. Furthermore, the trajectories have been numerically differentiated, smoothed, and normalized.

For the most part, the velocity norm generated from the trajectory provides sufficient information to discriminate between characters. However, there exist few combinations of characters that present difficult classification problems. To this end, we set up our experiment as binary classification of two characters with similar forms. For conciseness, we present the results of three pairs: “c/e”, “g/q”, and “u/v”, which of the possible combinations, were among the most difficult to classify.

The velocity norm were computed from the $x - y$ coordinates, and the subsequences of each time series were prepared with the same procedure described in the previous section. The subsequences of respective pairs are joined and referred to as $H1$, $H2$, and $H3$.

### 6.2 Experimental Setup

We evaluate the proposed algorithm by cross-validation. We perform a 5-fold cross-validation for the interaction datasets, and perform 3-fold cross-validation for the sensor tag and the handwriting datasets. For the pivot behavior data, we use each of T1-3 as one fold of a 3-fold cross-validation. Although the pivot behavior is concentrated to one segment of the time series, the subsequences are randomized so that the imbalance of the class does not affect the training. Table 1 summarizes the number of samples and the ratios of positive and negative examples in respective datasets.

In [10], the trajectories of the pivot behavior were analyzed using STS clustering. The subsequences of the velocity norm were generated by sliding window of three different sizes. We adopt them as the decision windows $\mathcal{K}$. For other datasets, three decision windows are chosen arithmetically. For the sake of comparison, we empirically confirmed that neither the 1-NN or the SVM classifier performs significantly worse than the other at each decision window. For all datasets, the weights are defined such that $\rho_k$ is the inverse squared proportion of $\lambda_k$ to the largest decision window, i.e.,

$$\rho_k = \left(\frac{\lambda_k}{\max \mathcal{K}}\right)^{-2}.$$  

Table 2 summarizes the setup of TSC for each dataset.

### 6.3 Evaluation Measure

The accuracy and the error rate are standard measures of performance for a classification problem. Additionally, if the classifier is allowed to reject, e.g., yield $y'(t) = 0$ for an inconclusive prediction, the rejection rate is also considered.

Given the true class $\mathcal{Y} = \{y_t\}$ and the predictions $\mathcal{Y}' = \{y'(t)\}$, the accuracy and the rejection rate are

$$\text{ACC} = \frac{\# \{t : y_t = y'(t)\}}{\# \{t \}}$$

The error rate is given by

$$\text{ERR} = 1 - (\text{ACC} + \text{RJT})$$

For TSC, we additionally consider weighted measures reflecting the severity of errors at each timing. Simply taking the inverses of $\mathcal{R}$ as the weight, the weighted contingency table is defined as follows.

$$Q = \begin{pmatrix}
q(1,1) & q(1,0) & q(1,-1) \\
q(-1,1) & q(-1,0) & q(-1,-1)
\end{pmatrix}$$

where $q_b = \sum_{t \in \{b = (y,y'(t))\}} p_A(t)$.

The weighted accuracy and error rate are defined as

$$\text{WAC} = \frac{q(1,1) + q(-1,-1)}{\eta}, \quad \text{WER} = \frac{q(1,-1) + q(-1,1)}{\eta}$$

In our experiment, we consider WAC and WER along with ACC, ERR, and RJT for performance evaluation.

### 6.4 Baseline Methods

We adapt two state-of-the-art methods in the related problems: Early Classification of Time Series [4] and Empirical Thresholding [14] to provide the baseline performances. ECTS is a hybrid instance-based learning developed for the early classification problem that combines hierarchical clustering and 1-NN classification. It can be directly applied to the TSC problem. Empirical Thresholding is a meta-learning algorithm for cost-sensitive learning. We use ET to choose thresholds of $k$ classifiers in the ensemble. As we choose the thresholds in a greedy manner from earlier to later, we refer to our adapted implementation as Greedy Empirical Thresholding. Further details are omitted due to space.

### 6.5 Results

In all trials, the proposed algorithm converged after few tens of iterations. Figures 3(a)-3(d) illustrates the standard performance measures on respective datasets. In each figure, the first, second, and third blocks of each stacked bar quantifies the accuracy, the rejection and error rates of a method, respectively. Table 3 summarizes the weighted performance measures.
Table 1: Dataset Summary

<table>
<thead>
<tr>
<th></th>
<th>T1</th>
<th>T2</th>
<th>T3</th>
<th>PX1</th>
<th>PX2</th>
<th>PX3</th>
<th>S1</th>
<th>S2</th>
<th>S3</th>
<th>S4</th>
<th>H1</th>
<th>H2</th>
<th>H3</th>
</tr>
</thead>
<tbody>
<tr>
<td>η</td>
<td>4000</td>
<td>3500</td>
<td>5000</td>
<td>6490</td>
<td>6440</td>
<td>6410</td>
<td>1036</td>
<td>939</td>
<td>932</td>
<td>575</td>
<td>984</td>
<td>786</td>
<td>858</td>
</tr>
<tr>
<td>T:F</td>
<td>1:2.3</td>
<td>1:6</td>
<td>1:43</td>
<td>1:43</td>
<td>1:43</td>
<td>1:43</td>
<td>1:4.0</td>
<td>1:3.2</td>
<td>1:4.2</td>
<td>1:2.8</td>
<td>1:76</td>
<td>1:1.1</td>
<td>1:85</td>
</tr>
</tbody>
</table>

Table 3: Weighted Performance Measures

<table>
<thead>
<tr>
<th></th>
<th>TSE</th>
<th>GET</th>
<th>ECTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>TSE</td>
<td>WAC</td>
<td>WER</td>
<td>WAC</td>
</tr>
<tr>
<td>T1-3</td>
<td>2.6</td>
<td>0.002</td>
<td>2.1</td>
</tr>
<tr>
<td>PX1</td>
<td>0.26</td>
<td>0.015</td>
<td>0.22</td>
</tr>
<tr>
<td>PX2</td>
<td>0.3</td>
<td>0.02</td>
<td>0.35</td>
</tr>
<tr>
<td>PX3</td>
<td>0.24</td>
<td>0.015</td>
<td>0.23</td>
</tr>
<tr>
<td>S1</td>
<td>0.26</td>
<td>0.024</td>
<td>0.25</td>
</tr>
<tr>
<td>S2</td>
<td>0.14</td>
<td>0.027</td>
<td>0.1</td>
</tr>
<tr>
<td>S3</td>
<td>0.25</td>
<td>0.054</td>
<td>0.27</td>
</tr>
<tr>
<td>S4</td>
<td>0.49</td>
<td>0.05</td>
<td>0.39</td>
</tr>
<tr>
<td>H1</td>
<td>2.3</td>
<td>0.024</td>
<td>2.3</td>
</tr>
<tr>
<td>H2</td>
<td>1.3</td>
<td>0.042</td>
<td>0.43</td>
</tr>
<tr>
<td>H3</td>
<td>1.4</td>
<td>0.048</td>
<td>1.9</td>
</tr>
</tbody>
</table>

Fig.3(a) shows TSE achieved the lowest error rate and the highest accuracy for the Pivot Behavior dataset. ECTS shows a higher accuracy than GET, but its WAC is lower as shown in Table 3, due to its larger decision length. GET is more prone to early errors and a larger WER. TSE also achieves the best WAC and WER.

In Fig.3(b), TSE exhibits significantly higher accuracies than ECTS, with marginally larger error rates. ECTS achieves the lowest error rate for PX1,2 and a trivially larger error rate than TSE for PX3. GET achieves high accuracies but even higher error rates, which suggests over-fitting at early decision windows. Comparing WAC and WER from PX1-3 in Table 3, GET is comparable to TSE with regards to WAC but has much higher WER. ECTS made many predictions at the final decision window, resulting in small WAC and WER.

From Fig.3(c), TSE achieved the highest accuracies for S1-4 and also the lowest error rates for S2-4. ECTS achieved generally comparable error rates, but made all predictions on the last decision window as apparent from its WAC and WER in Table 3. TSE achieves significantly higher WAC while achieving lower or comparable WER. GET achieves significantly higher error rates and WER than others.

Fig.3(d) shows that the TSE achieves much higher accuracies and comparable error rates for H1-3, compared to ECTS. GET achieves the best accuracies on H1 and H3 but with significantly larger error rates. In Table 3, TSE achieves significantly higher WAC than ECTS while WER is lower for H1 and comparable for H2,3. Most predictions of ECTS are made at the final
window. GET shows significantly large WER.

The overall tendency of ECTS can be described as prudent, exploiting long decision windows and making generally few errors, which resulted in low WAC and WER. The greedy selection of the threshold resulted in audacious predictions yielding high accuracies and even higher error rates. Its early classifiers were especially prone to over-fitting. TSE achieves better overall accuracy compared to the greedy threshold selection with significantly smaller error rates. It is also as cautious as ECTS with regards to the error rates, but achieves generally better accuracies. It makes predictions much earlier than ECTS, evidenced by its higher WAC.

7 Conclusion

In this paper, we addressed the classification problem of the subsequences of time series with a time-sensitive goal to predict the unobserved class of events early and accurately. We proposed an ensemble scheme to make predictions at varied timings and to commit to the earliest confident prediction. We presented a cutting-plane algorithm, which alternates between training the individual classifiers based on the errors among the committed predictions and choosing the timing of commitment while minimizing the joint error of the ensemble.

The empirical study of the time-sensitive classification problem showed that the proposed ensemble classifier outperforms the greedy cost-sensitive approach and a sophisticated nearest neighbor classifier for the Early Classification problem. It indicates that the difficulty of Time-sensitive Classification problem is of a different nature from that of Early Classification. While ECTS classifier is able to reduce the number of observations required to make the predictions at a constrained accuracy, it does not necessarily improve the classification performances at desired timings. Integrating varying costs and minimizing the losses at designated timings are thus important contributions of our work. For future work, it is important to analyze the sensitivity of TSE and how to adjusting the relative weight, $R$, for specific preferences.

References