Learning Future Classifiers without Additional Data

Atsutoshi Kumagai  
NTT Secure Platform Laboratories,  
NTT Corporation  
3-9-11, Midori-cho, Musashino-shi, Tokyo, Japan  
kumagai.atutoshi@lab.ntt.co.jp

Tomoharu Iwata  
NTT Communication Science Laboratories,  
NTT Corporation  
2-4, Hikaridai, Seika-cho, Soraku-gun, Kyoto, Japan  
iwata.tomoharu@lab.ntt.co.jp

Abstract

We propose probabilistic models for predicting future classifiers given labeled data with timestamps collected until the current time. In some applications, the decision boundary changes over time. For example, in spam mail classification, spammers continuously create new spam mails to overcome spam filters, and therefore, the decision boundary that classifies spam or non-spam can vary. Existing methods require additional labeled and/or unlabeled data to learn a time-evolving decision boundary. However, collecting these data can be expensive or impossible. By incorporating time-series models to capture the dynamics of a decision boundary, the proposed model can predict future classifiers without additional data. We developed two learning algorithms for the proposed model on the basis of variational Bayesian inference. The effectiveness of the proposed model is demonstrated with experiments using synthetic and real-world data sets.

1 Introduction

The decision boundary dynamically changes over time in some applications. For example, in spam mail classification, spammers continuously create new spam mails to overcome spam filters, and therefore, the decision boundary that classifies spam or non-spam can vary (Fdez-Riverola et al. 2007). In activity recognition using sensor data, the decision boundary evolves over time, the performance of the classifiers deteriorate quickly (Gama et al. 2014).

Many methods have been proposed for updating classifiers to maintain performance, such as online learning (Rosenblatt 1958; Crammer, Dredze, and Pereira 2009; Crammer, Kulesza, and Dredze 2009), forgetting algorithms (Klinkenberg 2004), time-window methods (Babcock et al. 2003; Kolter and Maloof 2007). These methods require labeled data to update a time-evolving decision boundary. However, it would be quite expensive to continuously collect labeled data since labels need to be manually assigned by domain experts. Methods that use only unlabeled data for updating have been proposed (Shimodaira 2000; Dyer, Capo, and Polikar 2014). However, using unlabeled data might be difficult. For example, in the security domain, anti-virus software vendors periodically distribute update files (that is, data for updating classifiers) to their users. Users employing anti-virus software in an offline environment, which carries a risk of virus infections via physical mediums such as USB, cannot receive these files and therefore cannot update their classifiers.

In this paper, we propose a novel approach for classifying data in the future when only labeled data with timestamps collected until the current time are available. With the proposed approach, a decision boundary is defined by the parameters of classifiers, and the dynamics of the decision boundary are modeled by using time-series models. We present a probabilistic model to realize this approach, where vector autoregressive and logistic regression models are used for the time-series model and classifier, respectively. Since our proposed model takes into account the time evolution of the decision boundary, the proposed method can reduce the deterioration of classification performance without additional data.

We developed two learning algorithms for the proposed model on the basis of variational Bayesian inference. The first algorithm is a two-step algorithm, which learns a time-series model after the classifiers learned. The second algorithm is a one-step algorithm, which learns a time-series model and classifiers simultaneously.

The remainder of this paper is organized as follows. In Section 2, we briefly review related work. We formulate the proposed model in Section 3 and describe two learning algorithms for the proposed model on the basis of variational Bayesian inference in Section 4. In Section 5, we demonstrate the effectiveness of the proposed method with experiments using synthetic and real-world data sets. Finally, we present concluding remarks and a discussion of future work in Section 6.

2 Related Work

To capture a time-evolving decision boundary, updating classifiers with labeled data has been widely used. Online learning algorithms have been developed to update classifiers after the arrival of labeled data (Rosenblatt 1958; Crammer, Dredze, and Pereira 2009; Crammer, Kulesza, and Dredze 2009). Forgetting algorithms learn the latest deci-
sion boundary by using weighting training data with their age. This weighting assumes that recent data are influential for determining the decision boundary. Examples of weighting include linear decay (Koychev 2000), exponential decay (Klinkenberg 2004), and time-window techniques that use only training data obtained recently (Babcock et al. 2002). Ensemble learning combines multiple classifiers to create a better classifier (Vapnik 1999; Schapire and Freund 2002; Brzezinski and Stefanowski 2014). Its mixture weight is learned so that classifiers that capture a current decision boundary well have a large weight. In some methods, active learning is a method for determine data which should be labeled is used for learning the time-evolving decision boundary (Zhu et al. 2010; Zliobaite et al. 2014). Methods for learning the boundary by using unlabeled data also have been proposed (Zhang, Zhu, and Guo 2009; Dyer, Capo, and Polikar 2014). All of these methods continuously require labeled and/or unlabeled data to capture the time-evolving decision boundary. However, the purpose of this study is to be labeled and/or unlabeled data to capture the time-evolving decision boundary in the future.

Our goal is to find classifiers \( \mathcal{H} = \{ h_1, \ldots, h_m \} \) for \( T \) future samples at time \( t \), which can precisely classify data at time \( t \) to time \( t+T \). We denote \( D_t := \{ (x_n, y_n) \}_{n=1}^N \) be a set of training data collected at time \( t \), where \( x_n \in \mathbb{R}^d \) is the \( d \)-dimensional feature vector of the \( n \)th sample at time \( t \), \( y_n \in \{ 0, 1 \} \) is its class label, and \( N_t \) is the number of training data collected at time \( t \). We discretize time at regular intervals, and data collected in the same time unit are regarded as data in the same time. We denote \( X_t := \{ x_n^t \}_{n=1}^{N_t} \) and \( Y_t := \{ y_n^t \}_{n=1}^{N_t} \). Our goal is to find classifiers \( \mathcal{H}_t : \mathbb{R}^d \rightarrow \{ 0, 1 \}, \ t \in \{ T+1, T+2, \ldots \} \), which can precisely classify data at time \( t \), given a set of training data \( D_t := \{ D_t \}_{t=1}^T \).

Next, we propose a probabilistic model for predicting future classifiers. Figure 1 illustrates our approach. Using labeled data from time 1 to \( T \), the decision boundary for each time, which is defined by classifier \( h_t \), and the dynamics of the decision boundary are learned. Then, the decision boundary in the future \( t = T + 1, T + 2, \ldots \) is predicted by using the learned decision boundary and dynamics.

We explain the proposed probabilistic model that realizes our approach. The proposed model assumes that the posterior probability of label \( y_n \) given feature vector \( x_n \) is modeled by logistic regression as.

\[
p(y_n^t = 1|x_n^t, \theta_t) = \sigma(w_n^\top x_n^t) = \frac{1}{1 + e^{-w_n^\top x_n^t}}, \quad (1)
\]

where \( w_t \in \mathbb{R}^d \) is a parameter vector of the classifier \( h_t \), \( \sigma(\cdot) \) is the sigmoid function, and \( \top \) denotes transposition. Note that the posterior probability that \( y_n^t = 0 \), \( p(y_n^t = 0|x_n^t, \theta_t) \), is equal to \( 1 - p(y_n^t = 1|x_n^t, \theta_t) \).

We capture the dynamics of the classifier parameter \( w_t \) by using the vector autoregressive (VAR) model, which is one of the classic time-series models (Lütkepohl 2005). The order \( p \) VAR model assumes that \( w_t \) depends linearly on the previous decision boundary \( w_{t-1}, \ldots, w_{t-m} \) as follows.

\[
w_t \sim \mathcal{N}(w_t; \sum_k A_kw_{t-k} + \theta_0^{-1}I_d), \quad (2)
\]

where \( \mathcal{N}(\mu, \Sigma) \) is a normal distribution with mean \( \mu \) and covariance matrix \( \Sigma \), \( A_1, \ldots, A_k \in \mathbb{R}^{d \times d} \) are \( d \times d \) matrices for defining dynamics, \( \theta_0 \in \mathbb{R}^d \) is a bias term, \( \theta \in \mathbb{R}^d \) denotes a hyper parameter, and \( I_d \) represents a \( d \times d \) identity matrix. In this paper, we restrict \( A_1, \ldots, A_k \in \mathbb{R}^{d \times d} \) to diagonal matrices for simplicity. This means the \( k \)th component of classifier parameter \( w_{t,i} \) depends only on its own previous parameter value \( w_{t-1,i}, w_{t-2,i}, \ldots \).

The proposed model assumes that the dynamics and bias parameters \( A_k \) and \( \theta_0 \) are generated from a Gamma distribution.

\[
\gamma_k \sim \text{Gam}(\alpha_k), \quad \theta_0 \sim \text{Gam}(\beta_0),
\]

where \( \text{Gam}(\alpha, \beta) \) is a Gamma distribution with \( \alpha \) and \( \beta \) parameters.

The joint distribution of labeled data \( D_t := \{ D_t \}_{t=1}^T \), classifier parameters \( W := \{ w_1, \ldots, w_T \} \), and bias parameters \( A := \{ A_0, A_1, \ldots, A_k \} \), precision parameters \( \Gamma := (\gamma_0, \gamma_1, \ldots, \gamma_m) \), \( \theta \) and \( \theta_0 \) is written as:

\[
p(D, W, A, \Gamma, \theta, \theta_0) = p(D|W)p(W|A, \theta, \theta_0)p(\theta|\Gamma)p(\Gamma)p(\theta)p(\theta_0)
\]

\[
= \prod_{t=1}^T \prod_{n=1}^{N_t} p(y_n^{t+1}|x_n^{t+1}, w_t) \cdot \prod_{t=1}^m \mathcal{N}(w_t|0, \theta_0^{-1}I_d)
\]
\[
\begin{align*}
&\times \prod_{t=m+1}^{T} \mathcal{N}\left( w_t \middle| \sum_{k=1}^{m} A_k w_{t-k} + A_0, \theta^{-1} I_d \right) \\
&\times \prod_{k=0}^{m} \mathcal{N}(A_k \mid 0, \gamma_k^{-1} I_d) \cdot \text{Gam}(\gamma_k | a_k, b_k) \\
&\times \text{Gam}(\theta | u, v) \cdot \text{Gam}(\theta_0 | u_0, v_0). \quad (3)
\end{align*}
\]

Since our model takes into account the time development of the classifier parameter, our method improves the classification performance in a non-stationary environment. Note that though we employ the logistic regression model and the vector autoregressive model for describing our proposed model, it is possible to use other models such as boosting (Freund, Schapire, and others 1996) and Gaussian processes (Roberts et al. 2013).

4 Inference

We developed two learning algorithms for the proposed model on the basis of variational Bayesian inference (Bishop 2006). The first algorithm is a two-step algorithm, which learns a time-series model after the classifiers learned. The second algorithm is a one-step algorithm, which learns a time-series model and classifiers simultaneously.

4.1 Two-step Learning Algorithm

We explain the two-step algorithm. First, classifier parameters \( w_t \) is learned for each time \( t \) by the maximizing a posterior estimation. Since dynamics parameters are unknown, we assume that the mean of the classifier parameters \( w_t \) is the parameters at the previous time \( w_{t-1} \), \( p(w_t | w_{t-1}, \theta) = \mathcal{N}(w_t | w_{t-1}, \theta^{-1} I_d) \). Then, the log posterior to be maximized is as follows.

\[
\begin{align*}
\log p(w_t | X_t, Y_t, w_{t-1}, \theta) &= \log p(Y_t | w_t, X_t) + \log p(w_t | w_{t-1}, \theta). \quad (4)
\end{align*}
\]

Classifier parameters \( W \) are obtained by repeating this procedure from \( w_1 \) to \( w_T \), where \( w_0 \) is a zero vector \( \mathbf{0} \).

Next, we learn the time-series model on the basis of variational Bayesian inference. Using learned classifier parameters \( W \) as observed data, we consider the following joint distribution instead of (3).

\[
\begin{align*}
p(W, A, \Gamma, \theta) &= p(W | A, \theta)p(A | \Gamma)p(\Gamma)p(\theta) \\
&= \prod_{t=m+1}^{T} \mathcal{N}(w_t \middle| \sum_{k=1}^{m} A_k w_{t-k} + A_0, \theta^{-1} I_d) \\
&\times \prod_{k=0}^{m} \mathcal{N}(A_k \mid 0, \gamma_k^{-1} I_d) \cdot \text{Gam}(\gamma_k | a_k, b_k) \cdot \text{Gam}(\theta | u, v). \quad (5)
\end{align*}
\]

We obtain an approximate posterior distribution \( q(A, \Gamma, \theta) \), called the variational posterior, of the posterior distribution \( p(A, \Gamma, \theta | W) \) since analytically calculating \( p(A, \Gamma, \theta | W) \) is impossible. It is known that the variational posterior \( q(A, \Gamma, \theta) \) is a local maximum point of the following lower bound.

\[
L(q) = \int q(A, \Gamma, \theta) \log \frac{p(W, A, \Gamma, \theta)}{q(A, \Gamma, \theta)} dA d\Gamma d\theta. \quad (6)
\]

In this paper, the variational posterior \( q(A, \Gamma, \theta) \) is assumed to factorize as follows.

\[
q(A, \Gamma, \theta) = \prod_{k=0}^{m} q(A_k)q(\gamma_k)q(\theta). \quad (7)
\]

By calculating the derivatives of the lower bound (6) with respect to \( q(A_k) \), \( q(\gamma_k) \) and \( q(\theta) \), we find that each distribution of (7) has the following functional form.

\[
\begin{align*}
q(\alpha_k) &= \mathcal{N}(\alpha_k | \mu_k, S_k^{-1}), \quad k = 0, \ldots, m, \\
q(\gamma_k) &= \text{Gam}(\gamma_k | a_k^\gamma, b_k^\gamma), \quad k = 0, \ldots, m, \\
q(\theta) &= \text{Gam}(\theta | u^\theta, v^\theta), \quad (8)
\end{align*}
\]

where \( \alpha_k \in \mathbb{R}^d \) is a diagonal vector \((A_k, \ldots, A_kd)\) of \( A_k \), \( \mu_k \in \mathbb{R}^d \), \( S_k \in \mathbb{R}^{d \times d} \), and \( a_k^{\gamma}, b_k^{\gamma}, u^\theta, v^\theta \in \mathbb{R}_+ \). These parameters are expressed as

\[
\begin{align*}
S_0 &= \left( \frac{a_0^\gamma}{b_0^\gamma} + (T - m) \frac{u^\theta}{v^\theta} \right) I_d, \\
\mu_0 &= S_0^{-1} \frac{u^\theta}{v^\theta} \sum_{t=m+1}^{T} \left( w_t - \sum_{k=1}^{m} \text{dg}(\mu_k)w_{t-k} \right), \\
S_k &= \frac{a_k^{\gamma}}{b_k^{\gamma}} I_d + \frac{u^\theta}{v^\theta} \sum_{t=m+1}^{T} \text{dg}(w_{t-k})^2, \\
\mu_k &= S_k^{-1} \frac{u^\theta}{v^\theta} \sum_{t=m+1}^{T} \left( w_t - \sum_{l \neq k} \text{dg}(\mu_l)w_{t-l} - \mu_0 \right) w_{t-k}, \\
&\quad \quad a_k^\gamma = a_k + \frac{1}{2} d, \quad b_k^{\gamma} = b_k + \frac{1}{2} \| \mu_k \|^2 + \text{Tr}(S_k^{-1}), \\
u^\theta = u + \frac{1}{2} (T - m)d, \\
v^\theta = v + \frac{1}{2} \sum_{t=m+1}^{T} \| w_t - \sum_{l=1}^{m} \text{dg}(\mu_l)w_{t-l} - \mu_0 \|^2 + \text{Tr}(S_0^{-1}) + \sum_{l=1}^{m} w_{t-l}^\top S_l^{-1} w_{t-l}, \quad (9)
\end{align*}
\]

where \( \text{dg}(x) \) returns a diagonal matrix whose diagonal elements are \( x = (x_1, \ldots, x_d), \| \cdot \| \) is \( \ell^2 \)-norm, and \( \text{Tr} \) represents the trace. In variational Bayesian inference, the variational posterior (7) is calculated by updating the variables in (8) by using equations (9) until some convergence criterion are satisfied.

4.2 One-step Learning Algorithm

With the one-step algorithm, we learn the time-series model and classifiers simultaneously.

The lower bound \( L(q) \) of the log marginal likelihood of the proposed model is expressed as:

\[
\begin{align*}
L(q) &= \int q(W, A, \Gamma, \theta, \theta_0) \log \frac{p(D, W, A, \Gamma, \theta, \theta_0)}{q(W, A, \Gamma, \theta, \theta_0)} dWdAd\Gamma d\theta d\theta_0. \quad (10)
\end{align*}
\]
We assume that the variational posterior \( q(W, A, \Gamma, \theta, \theta_0) \) is factorized as follows.

\[
q(W, A, \Gamma, \theta, \theta_0) = \prod_{t=1}^{T} \prod_{i=1}^{d} q(w_{t,i}) \prod_{k=0}^{m} q(A_k) q(\gamma_k) q(\theta) q(\theta_0).
\]  

We would like to maximize the lower bound (10) in a similar manner to the two-step algorithm. However, \( p(y_n^t | x_n^t, w_t) \) cannot be calculated analytically because of its nonlinearity. To deal with this problem, we use the following inequality based on (Jaakkola and Jordan 2000).

\[
p(y_n^t | x_n^t, w_t) \geq -e^{v_{\theta,0}} \sigma(\xi_n^t) \left( \frac{a + \xi_n^t}{2} + h(\xi_n^t) (a^2 - \xi_n^t) \right),
\]

where \( a := w_t^T x_n^t, \xi_n^t \in \mathbb{R} \) is a parameter that is associated with each training data \( (x_n^t, y_n^t) \) and determines the accuracy of the approximation, and \( h(\xi_n^t) := \frac{1}{2} \pi^2 \sigma(\xi_n^t - \frac{1}{2}) \). By substituting the term on the right side of (12) to the lower bound \( L(q) \), we obtain a new lower bound \( L(q, \xi) \). Note that since \( L(q) \geq L(q, \xi) \) for all \( q \) and \( \xi_n^t \), increasing the value of \( L(q, \xi) \) leads to an increased value of \( L(q) \). By calculating the derivatives of \( L(q, \xi) \) with respect to \( q(w_{t,i}), q(A_k), q(\gamma_k), q(\theta), q(\theta_0), \) and \( \xi_n^t \), we find that the variational posterior becomes as follows.

\[
q(w_{t,i}) = \mathcal{N}(w_{t,i} | \eta_{t,i}, \Lambda_{t-1}^{-1}), \quad t = 1, \ldots, T, \; i = 1, \ldots, d,
\]

\[
q(\alpha_k) = \mathcal{N}(\alpha_k | \mu_k, S_k^{-1}), \quad k = 0, \ldots, m,
\]

\[
q(\gamma_k) = \text{Gam}(\gamma_k | a_k^\gamma, b_k^\gamma), \quad k = 0, \ldots, m,
\]

\[
q(\theta) = \text{Gam}(\theta | u^\theta, v^\theta),
\]

\[
q(\theta_0) = \text{Gam}(\theta_0 | u^\theta_0, v^\theta_0),
\]

where \( \eta_{t,i}, \Lambda_{t,i}, a_k^\gamma, b_k^\gamma, u^\theta, v^\theta, u^\theta_0, v^\theta_0 \in \mathbb{R}, \mu_k \in \mathbb{R}^d \) and \( S_k \in \mathbb{R}^{d \times d} \). These parameters are determined by

\[
S_0 = \begin{pmatrix} a^\gamma_{00} & (T - m) a^0_{00} \\ b^0_{00} \end{pmatrix},
\]

\[
\mu_0 = S_0^{-1} \frac{u^\theta}{v^\theta} \sum_{t=1}^{T} \left( \eta_t - \sum_{k=1}^{m} \text{dg}(\mu_k) \eta_{t-k} \right),
\]

\[
S_k = \frac{a^\gamma_k}{b^\gamma_k} I_d + \frac{u^\theta}{v^\theta} \sum_{t=m+1}^{T} \text{dg}(\eta_{t-k})^2 + \Lambda_{t-1,k},
\]

\[
\mu_k = S_k^{-1} \frac{u^\theta}{v^\theta} \sum_{t=m+1}^{T} \text{dg} \left( \eta_t - \sum_{l \neq k} \text{dg}(\mu_l) \eta_{t-l} - \mu_0 \right) \eta_{t-k},
\]

\[
a_k^\gamma = a_k + \frac{1}{2} d, \quad b_k^\gamma = b_k + \frac{1}{2} \| \mu_k \|^2 + \text{Tr}(S_k^{-1}),
\]

\[
k = 1, \ldots, m,
\]

\[
u^0 = u + \frac{1}{2} (T - m)d,
\]

\[
u^\theta = v + \frac{1}{2} \sum_{t=m+1}^{T} \{ \text{Tr}(S_0^{-1} + \Lambda_{t-1}^{-1}) + \sum_{l=1}^{m} \eta_{t-l} S_{l-1}^{-1} \eta_{t-l} \}
\]

\[
+ \sum_{l=1}^{m} \text{Tr}(\text{dg}(\mu_l)^2 \Lambda_{l-1}^{-1} + \text{Tr}(S_{l-1}^{-1} \Lambda_{l-1}^{-1})},
\]

\[
u^\theta_0 = u_0 + \frac{1}{2} T m d,
\]

\[
u^\theta_0 = v_0 + \frac{1}{2} \sum_{t=1}^{T} \| \eta_t \|^2 + \text{Tr}(\Lambda_{t-1}^{-1}),
\]

\[
\Lambda_t = \frac{v^\theta}{v^\theta_0} \sum_{l=1}^{K_t^1} \{ (y_n^t - 1/2 x_n^t) \},
\]

\[
\eta_{t,i} = \lambda_{t,i}^{-1} \left( \frac{\nu^\theta}{v^\theta} \sum_{l=1}^{K_t^1} (\eta_{l+1,i} - \mu_{0,i}) \mu_{l+1-i,i},
\]

\[- \frac{\nu^\theta}{v^\theta} \sum_{l=1}^{K_t^1} \sum_{l' \neq l} \eta_{l' + 1,i} - \mu_{l',i} \mu_{l+1-i,i},
\]

\[- \frac{\nu^\theta}{v^\theta} \sum_{l=1}^{K_t^1} \sum_{l' \neq l} \sum_{n=1}^{N_t} \{ (y_n^t - 1/2 x_n^t) \},
\]

\[
\Lambda_t = \frac{v^\theta}{v^\theta_0} \sum_{l=1}^{K_t^2} \{ (y_n^t - 1/2 x_n^t) \},
\]

\[
\eta_{t,i} = \lambda_{t,i}^{-1} \left( \frac{\nu^\theta}{v^\theta} \sum_{l=1}^{K_t^2} (\eta_{l+1,i} - \mu_{0,i}) \mu_{l+1-i,i},
\]

\[- \frac{\nu^\theta}{v^\theta} \sum_{l=1}^{K_t^2} \sum_{l' \neq l} \eta_{l' + 1,i} - \mu_{l',i} \mu_{l+1-i,i},
\]

\[- \frac{\nu^\theta}{v^\theta} \sum_{l=1}^{K_t^2} \sum_{l' \neq l} \sum_{n=1}^{N_t} \{ (y_n^t - 1/2 x_n^t) \},
\]

\[\xi_n^t = \alpha_n^T \left( \Lambda_{t-1}^{-1} + \eta_n^\top \eta_n \right) x_n^t, t = 1, \ldots, T, \; n = 1, \ldots, N_t
\]

where \( \eta_t := (\eta_{t,1}, \ldots, \eta_{t,d}), \mu_t := (\mu_{t,1}, \ldots, \mu_{t,d}), \Lambda_t := \text{dg}(\Lambda_{t,1}, \ldots, \Lambda_{t,d}), K_1^1 := \min(t, T - m), \) and \( K_2^1 := \min(m, T - t) \). We calculate the variational posterior (11) by updating the variables in (13) in turn via the equations (14) until some convergence criterion is satisfied.

Finally, we can predict the classifier parameters \( w_t \) for \( t = T + 1, T + 2, \ldots \) using the learned VAR model (2).
5 Experiments

We conducted experiments using two synthetic and two real-world data sets to confirm the effectiveness of the proposed method.

5.1 Synthetic Data

We used two synthetic data sets. A sample \((x_1, x_2) \in \mathbb{R}^2\) with label \(y \in \{0, 1\}\) and time \(t\) was generated from the following distribution \(p(x|y, t)\).

\[
\begin{align*}
x_1 &= 4.0 \cdot \cos(\pi(t/r + 1 - y)) + \epsilon_1, \\
x_2 &= 4.0 \cdot \sin(\pi(t/r + 1 - y)) + \epsilon_2,
\end{align*}
\]

where \(\epsilon_i\) for \(i = 1, 2\) is a random variable with a standard normal distribution. We can generate various data sets by changing the value of \(r\). Note that the data sets generated by (15) have periodicity (a period is equal to \(2r\)). \textit{Gradual} is a data set obtained with \(r = 20\). The decision boundary of this data gradually changed. \textit{Drastic} is a data set obtained with \(r = 2\). The decision boundary of this data drastically changed. In this experiments, we changed \(t\) from 1 to 20 and generated 100 samples for each label \(y\) and each time \(t\).

5.2 Real-world Data

We used two real-world data sets: \textit{ELEC2} and \textit{Chess.com}. Both data sets are public benchmark data sets for evaluating stream data classification or concept drift. \textit{ELEC2} contains 45312 instances and has eight features. \textit{Chess.com} contains 573 instances and has seven features. For details on these data sets, refer to the paper (Gama et al. 2014). In \textit{ELEC2}, we removed instances that have missing values since missing values are not the focus of our task. We removed instances with draw labels in \textit{Chess.com}. As a result, there were 27549 instances for \textit{ELEC2} and 503 instances for \textit{Chess.com}.

We used two synthetic data sets. A sample \((x_1, x_2) \in \mathbb{R}^2\) with label \(y \in \{0, 1\}\) and time \(t\) was generated from the following distribution \(p(x|y, t)\).

\[
\begin{align*}
x_1 &= 4.0 \cdot \cos(\pi(t/r + 1 - y)) + \epsilon_1, \\
x_2 &= 4.0 \cdot \sin(\pi(t/r + 1 - y)) + \epsilon_2,
\end{align*}
\]

where \(\epsilon_i\) for \(i = 1, 2\) is a random variable with a standard normal distribution. We can generate various data sets by changing the value of \(r\). Note that the data sets generated by (15) have periodicity (a period is equal to \(2r\)). \textit{Gradual} is a data set obtained with \(r = 20\). The decision boundary of this data gradually changed. \textit{Drastic} is a data set obtained with \(r = 2\). The decision boundary of this data drastically changed. In this experiments, we changed \(t\) from 1 to 20 and generated 100 samples for each label \(y\) and each time \(t\).

5.3 Setting

In our experiments, to find time variations in the performance of each method, we evaluated AUC by using data until a certain time unit for training, and the remaining for testing. For synthetic data sets, we created ten different sets every synthetic data set. We set \(T = 10\) and the remaining ten time units as test data. For \textit{ELEC2}, we set two weeks as one time unit, and then \(T = 29\) and the remaining ten time units as test data. For \textit{Chess.com}, we set 20 games as one time unit, \(T = 15\) and the remaining ten time units for test data. For the real-world data sets, we chose 80\% of instances randomly every time unit to create different training sets (five sets) and evaluated the average AUC by using these sets.

To demonstrate the effectiveness of the proposed method, we compared it with several methods: online logistic regression \textit{(online)}, batch logistic regression \textit{(batch)} and present logistic regression \textit{(present)}. \textit{Batch} learns a classifier by using all training data \(D\) at once. \textit{Online} learns a classifier by maximizing the log posterior (4) in turn from \(w_1\) to \(w_T\). \textit{Present} is a method that learns a classifier with only recent training data \(D_T\). We chose the regularization parameter for these three methods from \(\{10^{-1}, 1, 10^1\}\) in terms of which average AUC over the entire test periods was the best. With the proposed method, we set the hyperparameters as \(a_k = u = v_0 = 1, b_k = v = v_0 = 0.1\) for all data sets and fixed \(m\) as 3 for the real-world data sets from preliminary experiments. In addition, the regularization parameter for the two-step algorithm was set to the same value for \textit{online}.

5.4 Results

Table 1 shows averages and standard errors of AUC over the entire test time units for all data sets. For all data sets, the proposed method (especially \textit{one-step}) achieved the highest AUC. This result means that the proposed method captured...
Figure 3: Average and standard error of AUC with the proposed method for each test time unit over ten runs.

(a) two-step learning algorithm
(b) one-step learning algorithm

Figure 4: Average and standard error of AUC for each test time unit over five runs.

(a) ELEC2
(b) Chess.com

Figure 5: Average and standard error of AUC by the proposed method with different numbers of training data.

We compared the proposed two-step algorithm and proposed one-step one with ELEC2. Figure 5 shows the averages and standard errors of AUC with different numbers of training data at time $t$. Here, we used the same hyperparameters used before. One-step provided better classification performance than did two-step when the number of training data was small. Since two-step learns the time-series model after the classifiers learned, the parameters of the time-series model are sensitive to the learned classifiers, which are difficult to learn with a small number of training data. In comparison, one-step learned the classifiers and a time-series model simultaneously and therefore become more robust.

6 Conclusion and Future Work

We proposed probabilistic models for predicting future classifiers given labeled data with timestamps collected until the current time. We developed two learning algorithms for the proposed model on the basis of variational Bayesian inference. In experiments, we confirmed that the proposed method can reduce the deterioration of the classification ability better compared with existing methods. As future work, we will apply a non-linear time-series model to the proposed framework for data’s dynamics is non-linear.
References


Gama, J.; Žliobaitė, I.; Bifet, A.; Pechenizkiy, M.; and Bouchachia, A. 2014. A survey on concept drift adaptation. ACM Computing Surveys (CSUR) 46(4):44.


