Prediction and Change Detection In Sequential Data for Interactive Applications

Jun Zhou and **Li Cheng**
National ICT Australia
Locked Bag 8001, Canberra ACT 2601, Australia
{jun.zhou, li.cheng}@nicta.com.au

Walter F. Bischof
Department of Computing Science
University of Alberta, Canada
Edmonton, Alberta T6G 2E8
wfb@ualberta.ca

Abstract

We consider the problems of sequential prediction and change detection that arise often in interactive applications: A semi-automatic predictor is applied to a time-series and is expected to make proper predictions and request new human input when change points are detected. Motivated by the Transductive Support Vector Machines (Vapnik 1998), we propose an online framework that naturally addresses these problems in a unified manner. Our empirical study with a synthetic dataset and a road tracking dataset demonstrates the efficacy of the proposed approach.

Introduction

Consider the following scenario: You are working with a semi-automatic system, in a sequential manner, on a time-series (on tasks such as classification, regression, etc.) Once in a while during the process, the system may decide that it does not work properly – eg. it detects abrupt changes in the time-series – and it requests your input. The predictor then re-adapts itself, and the processing continues. This scenario characterizes a range of interactive applications, from interactive road tracking (*e.g.* Rochery, Jermyn, & Zerubia; Hu *et al.*; Zhou, Cheng, & Bischof), interactive video segmentation (*e.g.* Wang *et al.*) to web-based recommendation systems (*e.g.* Resnick & Varian). On the one hand, it is important to retain the "human-in-the-loop" in the sense that the predictor is able to request necessary guidance at change points. On the other hand, it is desirable to have as few such interventions as possible, as it is costly to ask the oracle.

There have been many studies on related tasks including designing and analyzing sequential prediction (Littlestone & Warmuth; Littlestone; Vovk; Kivinen, Smola, & Williamson) and change point detection (Basseville & Nikiforov; Kifer, Ben-David, & Gehrke; Ho). Despite the series of practical applications, work on this problem has been very limited.

In this paper, we propose a novel approach to address this problem from an online learning perspective. The approach is intuitive and flexible, and the learning rate (*i.e.,* step size) of the proposed online learning algorithm is self-adaptive. We also discuss related issues, such as dealing with non-stationary distributions, the issue of unbalanced data, and working with an ensemble of models. Our proposed work is motivated by transductive support vector machines (TSVM Vapnik) for semi-supervised learning, and it is also closely related to work on online learning (Littlestone & Warmuth; Kivinen, Smola, & Williamson) and on the detection of abrupt changes in time-series (Basseville & Nikiforov).

The paper is organized as follows: We start with the problem formulation, which leads to the proposed approach. A number of related issues are discussed, and by addressing them, we explain the details of the proposed algorithm. The applicability of the proposed approach is demonstrated with experiments on a synthetic simulation (interactive classification) and on a real-world problem (interactive road tracking).

The Approach

Problem Formulation

Let x denote an instance and let y be a label. An example contains either a pair (x, y) or only an instance x. An interactive application takes a time-series $\mathcal{T}$ as input and operates in sessions, $\mathcal{T} = (\cdots, S_i, \cdots)$. Each session $S = \Big((x_{j+1}, y_{j+1}), \cdots, (x_{j+m}, y_{j+m}), x_{j+m+1}, \cdots, x_{j+n}\Big)$, corresponds to one disjoint segment of the time-series, which starts with a few examples with corresponding human inputs, then delivers predictions through a sequence of unlabeled examples, and ends with an abrupt change. This change, which is detected by the session predictor h, triggers a request for new human input for the next few examples, leading to a new session. An example of performing interactive classification on an synthetic time-series is illustrated in Figure 2 top. The prediction problem could be one of classification, regression, ranking, or others, depending entirely on the interactive application at hand.

To begin with, let us consider a special case where the time-series contains exactly one session, i.e. $\mathcal{T} = \Big((x_1, y_1), \cdots, (x_m, y_m), x_{m+1}, \cdots, x_n\Big)$. This is closely related to the typical setting of semi-supervised learning[1],

[1] In case of classification (or regression), it is exactly a semi-supervised classification (or regression) problem.

where a large number of unlabeled examples is expected to help elucidate prediction, together with a few labeled examples. As advocated by (Vapnik 1998) in the TSVM, this can be achieved by exploiting the so-called cluster assumption: the prediction hyperplane should maintain a large margin over the dataset including both the labeled and unlabeled examples, and minimize a regularized risk function on the *entire* set of examples

$$\min_f \quad \frac{1}{2}\|f\|^2 + \lambda_l \sum_{i=1}^{m} L_l(x_i, y_i, f) + \lambda_u \sum_{j=m+1}^{n} L_u(x_j, f) \tag{1}$$

where $\lambda_l, \lambda_u > 0$ and f is a parameter vector to predict a label. In the case of binary classification, $h(x, f) = +1$ if $f(x) > 0$ and $h(x, f) = -1$ otherwise. We obtain a TSVM by letting $L_l(x_i, y_i, f) = (\rho_l - y_i f(x_i))_+$ and $L_u(x_j, f) = (\rho_u - |f(x_j)|)_+$, where the margins $\rho_l, \rho_u > 0$ and $(\cdot)_+ = \max\{0, \cdot\}$. In the following, we assume $\rho \triangleq \rho_l = \rho_u$. The induced optimization problem cannot be solved easily because the second loss term $(\rho - |f(x_j)|)_+$ is a non-convex function. A lot of effort has been devoted to minimizing non-convex objective functions (using, *e.g.* deterministic annealing or CCCP). Returning to our situation, things are even worse because there are multiple sessions, and, in addition to making predictions, abrupt changes have to be detected since we do not know a priori when a session should end.

In this paper, we consider an online learning framework, where, at time t, given the current parameter f_t and an example (x_t, y_t) (or x_t), we update the parameter f_{t+1} by minimizing a regularized risk function on the *current* example

$$f_{t+1} \;=\; \operatorname*{argmin}_f \;\; \frac{1}{2}\|f - f_t\|^2 + \eta_t L_l(x_t, y_t, f), \tag{2}$$

when (x_t, y_t) is presented. When only x_t is presented, this becomes

$$f_{t+1} \;=\; \operatorname*{argmin}_f \;\; \frac{1}{2}\|f - f_t\|^2 + \eta_t L_u(x_t, f). \tag{3}$$

In both cases, the new parameter f_{t+1} is expected to be reasonably close to the previous f_t (the first term), while incurring a small loss on the current example (the second term). η_t is a trade-off parameter that balances between the two objectives and is usually fixed a priori, i.e. $\eta = \eta_t,\ \forall t$.

There are distinct advantages to choose this online learning framework: First, it is computationally more efficient, and second, an online algorithm can be elegantly extended to track a slowly drifting target over time in one segment, as will be shown later. Moreover, as shown next, by incorporating the change detection component, we end up working with a convex objective function.

This framework is very flexible in the sense that various loss functions can be deployed for different applications. To illustrate this point, we present three types of loss functions:

Binary classification loss: We use binary hinge loss (Schölkopf & Smola 2002), which gives $L_l(x_t, y_t, f) = (\rho - y_t f(x_t))_+$ and $L_u(x_t, f) = (\rho - |f(x_t)|)_+$. This loss is used in applications such as interactive road tracking and video segmentation.

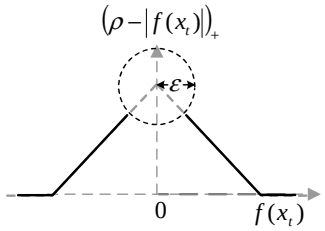

Figure 1: The figure illustrates the effect of considering change detection and classification problems together. For a given x_t, the horizontal axis denotes $f(x_t)$ and the vertical axis is the instantaneous loss. In classification, the loss function contains both the solid and the dashed lines, leading to a non-convex problem. When dealing with both change detection and classification, only the solid lines outside the ϵ-ball are left, which gives a convex problem.

Regression loss: Using insensitive loss (Schölkopf & Smola 2002), we have $L_l(x_t, y_t, f) = (|f(x_t) - y_t| - \rho)_+$ and $L_u(x_t, f) = (|f(x_t) - x_t| - \rho)_+$.

Ranking loss: We use ordinal regression hinge loss (Chu & Keerthi 2005) for ranking problems. Each instance x_t is associated with a vector $y \in \{-1, +1\}^r$ as follows: If the rank of an instance is k, we set the first k components of y to $+1$ and the rest of the components to -1. Now $f(x_t)$ is r-dimensional with the k-th component being $f(x_t, k)$. Then $L_l(x_t, y_t, f) = \sum_{k=1}^{r} (\rho - y_{t,k} f(x_t, k))_+$ and $L_u(x_t, f) = \sum_{k=1}^{r} (\rho - |f(x_t, k)|)_+$. This loss can be used in weblog-based recommendation systems.

Change Detection

We use a moving-average method, as described in (Basseville & Nikiforov 1993), to detect the change points in a classification problem. This is executed by maintaining the recent values of $\{x_a : a \in \mathcal{A}_c\}$ in a FIFO queue of fixed size $|\mathcal{A}|$ for class c. A change is detected if the distance between $\sum_a x_a / |\mathcal{A}|$ and x_t exceeds a threshold $\delta > 0$, when x_t is predicted as belonging to class c. This method can be easily extended to regression and ranking problems.

In addition, the cluster assumption also suggests that encountering a severely in-separable example (*i.e.,* it is too close to the prediction hyperplane in the case of classification and ranking) indicates the beginning of a new session. For a real $\epsilon > 0$, the predictor decides whether to request human input as follows: For classification and ranking problems, a change point is detected if $|f(x_t)| \leq \epsilon$ with $0 < \epsilon < \rho$, while, for regression, the rule becomes $|f(x_t) - x_t| \geq \epsilon$ where $0 < \rho < \epsilon$.

Having incorporated change detection in this manner, we now deal with a convex minimization problem. The reason is illustrated in Figure 1 on a classification problem, which was originally a non-convex problem due to the term $|f(x)|$, which peaks at zero. We use an ϵ-ball centered around the peak point, and the predictor stops asking for labels, whenever the value of $f(x)$ falls into the ϵ-ball, turning (3) into a convex minimization problem with the feasible regions being entirely outside the ϵ-ball. This holds similarly for regression and ranking problems. In addition, the following

lemma ensures that it is safe to proceed by just considering whether $f_t(x_t)$ is outside the ϵ-ball, which is much easier than to compute $f(x_t)$.

Lemma 1 (ϵ-ball)**.** *Let us assume $\tau = 0$ and consider the binary classification problem. If $f_t(x_t)$ is outside the ϵ-ball (*i.e., *either $f_t(x_t) < -\epsilon$ or $f_t(x_t) > \epsilon$), it is sufficient to guarantee that $f_{t+1}(x_t)$ is also outside the ϵ-ball.*

Proof. When $\tau = 0$, we know $f_{t+1}(x_t)$ can be represented as $f_t(x_t) + \alpha_t x_t$. $\epsilon < f_{t+1}(x_t) < \epsilon$ gives $\frac{-\epsilon - f_t(x_t)}{\langle x_t, x_t \rangle} < \alpha_t < \frac{\epsilon - f_t(x_t)}{\langle x_t, x_t \rangle}$. Now if we know that $f_t(x_t) > \epsilon$, then it gives $\alpha_t < 0$ as $\frac{\epsilon - f_t(x_t)}{\langle x_t, x_t \rangle} < 0$, which is clearly impossible, since $\alpha_t = \alpha_t^+ \geq 0$ as in (9). Similarly, if $f_t(x_t) < -\epsilon$, then it gives $\alpha_t > 0$ as $\frac{-\epsilon - f_t(x_t)}{\langle x_t, x_t \rangle} > 0$, which is also impossible, since $\alpha_t = \alpha_t^- \leq 0$ as in (10). □

Dealing with Non-stationary Distributions

So far we have considered the stationary scenario where, in each session of the time-series, the examples are drawn in hindsight from the same distribution. In practice, however, the examples in each session might drift slowly. This can be accommodated by extending (2) and (3) as follows:

$$f_{t+1} = \operatorname*{argmin}_f \ \frac{1}{2}\|f - f_t\|^2 + \left(\frac{\lambda}{2}\|f\|^2 + \mathrm{c}_l \mathrm{L}_l(x_t, y_t, f)\right), \tag{4}$$

and

$$f_{t+1} = \operatorname*{argmin}_f \ \frac{1}{2}\|f - f_t\|^2 + \left(\frac{\lambda}{2}\|f\|^2 + \mathrm{c}_u \mathrm{L}_u(x_t, f)\right), \tag{5}$$

where η_t is incorporated into λ and c_l (or c_u). This leads to a geometrical decay of previous estimates with $f_t = (1-\tau)f_{t-1} - c(1-\tau)\partial_f \mathrm{L}$, where $\tau = \frac{\lambda}{1+\lambda}$ and L is either $\mathrm{L}_l(x_t, y_t, f)$ or $\mathrm{L}_u(x_t, f)$.

Kernels

To make use of the powerful kernel methods, the proposed framework can be lifted to reproducing kernel Hilbert space (RKHS) $\mathcal{H}$ by letting $f \in \mathcal{H}$, with the defining kernel $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ satisfying the reproducing property, $\langle f, k(x, \cdot) \rangle_{\mathcal{H}} = f(x)$. The representer theorem (Schölkopf & Smola 2002) guarantees that f can be expressed uniquely as $f_t = (1-\tau)f_{t-1} + \alpha_t k(x_t, \cdot)$ (Cheng *et al.* 2006).

The Algorithm

We can now present our algorithm in detail. For ease of exploration, we focus only on the problem of binary classification. With proper modifications, it can be easily adapted to regression, ranking and other problems. Recall that, at time t, we are presented with an example x_t and possibly with a ground-truth label y_t, and we want to update the parameter f_t to f_{t+1} by incorporating the new example.

On the one hand, when (x_t, y_t) is presented, f is updated by solving the optimization problem (4) with $\mathrm{L}_l(x_t, y_t, f) = (\rho - y_t f(x_t))_+$. After simple derivations, we have $f_{t+1} = (1-\tau)f_t + \alpha_t k(x_t, \cdot)$, where

$$\alpha_t = \begin{cases} \hat{\alpha}_t & \text{if } y_t\hat{\alpha}_t \in [0, (1-\tau)c_l]; \\ 0 & \text{if } y_t\hat{\alpha}_t < 0; \\ y_t(1-\tau)c_l & \text{if } y_t\hat{\alpha}_t > (1-\tau)c_l, \end{cases} \tag{6}$$

with

$$\hat{\alpha}_t = \frac{\rho - (1-\tau)y_t f_t(x_t)}{y_t k(x_t, x_t)}.$$

On the other hand, when we have access only to x_t, f is updated by solving the optimization problem (5) with $\mathrm{L}_u(x_t, f) = (\rho - |f(x_t)|)_+$. As a result, the value of α_t is hinged on $f_t(x_t)$ and has two cases:

$$\alpha_t = \begin{cases} \alpha_t^+ & \text{if } f_t(x_t) \geq \epsilon; \\ \alpha_t^- & \text{if } f_t(x_t) \leq -\epsilon. \end{cases} \tag{7}$$

In case 1, lettting

$$\hat{\alpha}_t^+ = \frac{\rho - (1-\tau)f_t(x_t)}{k(x_t, x_t)}, \tag{8}$$

we have

$$\alpha_t^+ = \begin{cases} \hat{\alpha}_t^+ & \text{if } \hat{\alpha}_t^+ \in [0, (1-\tau)c_u]; \\ (1-\tau)c_u & \text{if } \hat{\alpha}_t^+ > (1-\tau)c_u; \\ 0 & \text{if } \hat{\alpha}_t^+ < 0. \end{cases} \tag{9}$$

Similarly, in case 2, letting

$$\hat{\alpha}_t^- = \frac{-\rho - (1-\tau)f_t(x_t)}{k(x_t, x_t)}, \tag{10}$$

we have

$$\alpha_t^- = \begin{cases} \hat{\alpha}_t^- & \text{if } \hat{\alpha}_t^- \in [-(1-\tau)c_u, 0]; \\ -(1-\tau)c_u & \text{if } \hat{\alpha}_t^- < -(1-\tau)c_u; \\ 0 & \text{if } \hat{\alpha}_t^- > 0. \end{cases} \tag{11}$$

Dealing with Unbalanced Data

For practical interactive classification applications, the number of examples are often unbalanced across different categories. In road tracking, for example, the number of positive examples (road examples) is much smaller than the number of negative ones (off-road examples). Our framework can be extended to deal with this issue. Consider a binary classification case, and let ρ^+ and ρ^- be the margins for the positive and negative sides of the separating hyperplane, respectively. When receiving (x_t, y_t), the loss is associated with proper margin conditioning on whether y_t is positive or negative. Similarly, when presented only with x_t, it is conditioned upon $f_t(x_t)$ as $L_u(x_t, f) = (\rho^+ - f(x_t))_+$ if $f_t(x_t) > 0$, and $L_u(x_t, f) = (\rho^- + f(x_t))_+$ otherwise.

An Ensemble of Predictors

Over time, the interactive system collects a number of predictors from past sessions. Intuitively this ensemble can help to improve predictions for new sessions. While sophisticated algorithms with guaranteed theoretical bounds exist

Algorithm 1 Classification and Change Detection

Input: The cut-off value c_l and c_u, the change detection threshold ϵ, the decay rate τ, the moving-average threshold δ.
Output: the set of predictions $\{\hat{y}\}$ and sessions $\{i\}$.
Initialize the time-series index $j \leftarrow 0$, session $i \leftarrow 0$, and initial parameter $f \leftarrow 0$.
repeat
  % — in session i —
  session index $t \leftarrow 0$
  while Receive (x_t, y_t) **do**
    update parameter f_{t+1} by computing α_t from (6)
    $t \leftarrow t+1, j \leftarrow j+1$
  end while
  % now receive only x_t
  while Receive x_t **do**
    if $|f_t(x_t)| < \epsilon$ or $|f_t(x_t) - \sum_{a \in |\mathcal{A}|} f_a(x_a)| > \delta$, then **break**
    Predict $\hat{y}_t$
    update parameter f_{t+1} by computing α_t from (7)
    $t \leftarrow t+1, j \leftarrow j+1$
  end while
  % a change is detected
  $i \leftarrow i+1$
until $j >= |\mathcal{T}|$

for ensemble methods (Dietterich 2000), we use a simple strategy: Recent predictors are maintained in a queue of bounded size, and when an abrupt change is detected, we first search in the queue for an optimal predictor that can still perform well on the change point and switch to this predictor to continue with automatic predictions rather than resorting to human input. The intuition is simple: the models learned in the past might turn out to be useful for the current scenario. This strategy is applied later in the road tracking application, improving the overall performance.

Experiments

We applied the proposed approach to the problem of interactive classification of a synthetic time-series and to a real-world road tracking task. The comparison approach is a nearest-neighbor moving average (NNMA) algorithm: In each session, the NNMA assigns to the current example a label according to the closest match among the human inputs, and change points are detected using the same moving average method. The parameters were tuned individually for good performance.

Performance Evaluation

The performance of an interactive system is usually evaluated objectively based on two criteria: accuracy and efficiency. The accuracy criterion mainly considers tracking errors (*i.e.,* those with large deviation from corresponding manual labels), while the efficiency criterion deals with the time saving for the human operator (*e.g.* by measuring how many mouse clicks the operator has made in a fixed set of road maps for interactive road tracking). These lead to a

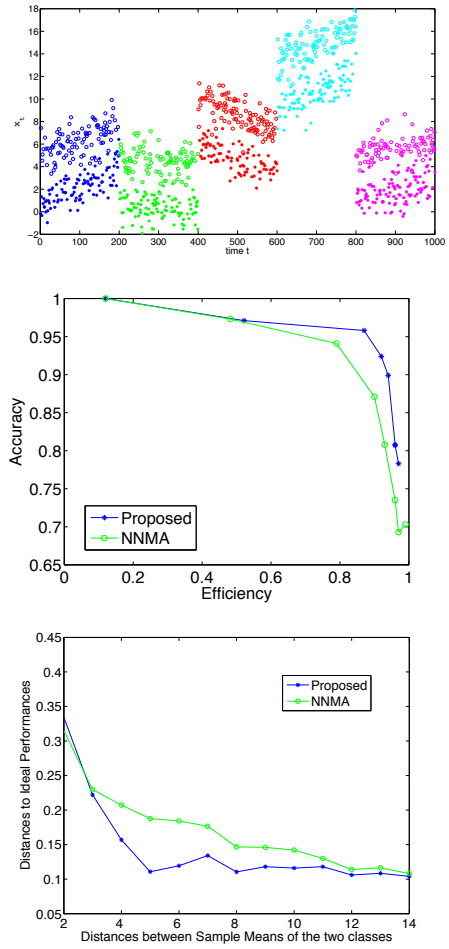

Figure 2: An interactive classification task on a synthetic time-series. Top: An exemplar synthetic time-series. Middle: A comparison of the proposed method and NNMA on the accuracy-efficiency plot. Bottom: A comparison of two methods over different levels of difficulty. See text for details.

2-D accuracy-efficiency plot (*e.g.* Figure 2 middle) where the horizontal axis measures accuracy and the vertical axis shows efficiency. After proper normalization, the results fall in a $[0, 1]$ bounding box. Similar to the ROC curve, the performance of an ideal system would fall in the top-right area of the box. We use this method to evaluate related systems throughout the experiments.

Interactive Classification on Synthetic Time-series

In this section, we present two experiments on a synthetic time-series. Figure 2 top presents an example time-series, which contains 1-D examples sampled with uniform probabilities from two classes (marked with circle an star signs). To mimic a real-world situation, each class-conditioned distribution, a 1-D Gaussian distribution, is allowed to drift slowly over the time. Five disjoint subsequences (marked with different colors) are sampled in this way, each using a distinct drifting pattern, to model abrupt changes. With this type of time-series, a semi-automatic system is expected to

predict with good accuracy and to make minimum queries for inputs.

Figure 2 middle shows a comparison of the NNMA method and the proposed approach on the accuracy-efficiency plot, where the results are averaged over five time-series. The results indicate that the proposed approach delivers better performance when the class-conditioned distributions drift over time, as is typical in real-world settings.

To get an idea of the robustness of the proposed approach, we conducted a second experiment, where the algorithms were evaluated on time-series of different levels of difficulty: At the easiest level, the means of the two class-conditional distributions are well separated, while at the most difficult level, the two means are very close to each other, with the standard deviations fixed throughout this experiment. In Figure 2 bottom, from left to right along the horizontal axis, the problems become easier to deal with as the gap between the sample means of both positive and negative classes grows. The vertical axis measures the distance to the ideal performance at the top-left corner (1,1) of the accuracy-efficiency plot. A small value therefore indicates better performance. Figure 2 bottom displays the results, where each value along the curves is computed by averaging over ten time-series. We can see that the proposed method gives overall better performance than NNMA.

Interactive Road Tracking (IRT)

IRT refers to a semi-automatic image understanding system to assist a cartographer annotating road segments in aerial photographs. Given an aerial photograph containing a number of roads, the IRT system assists the cartographer to sequentially identify and extract these road segments (including *e.g.* transnational highways, intrastate highways, and roads for local transportation). As shown in Figure 3 top, road-tracking is not a trivial task because road features vary considerably due to changes in road material, occlusions of the road, and a lack of contrast against off-road areas. It is extremely difficult for a fully automatic system to annotate the road segments with a reasonable accuracy. Much research has been devoted to road tracking from aerial photographs (*e.g.* Merlet & Zerubia; Geman & Jedynak; Geman & Jedynak; Yuille & Coughlan; Lacoste, Descombes, & Zerubia), and the attempts have been devoted to automatic systems. People have gradually realized that the human cartographer can and should provide help in these systems (*e.g.* Geman & Jedynak). In recent years, a number of semi-automatic systems have been proposed (Rochery, Jermyn, & Zerubia; Hu *et al.*; Zhou, Cheng, & Bischof).

In our system, a session comprises an *input* phase and a *detection and prediction* (DP) phase. The input phase contains a series of labeled examples along a road segment, where each example (also called an on-road profile) records the location and direction of the local road segment, together with features that characterize the local geometric texture. We also collect a set of off-road profiles by randomly sampling nearby regions. During the DP phase, the system searches ahead and returns a set of candidate profiles based on the location and direction of current road segment. Then the online predictor is applied to pick on-road profiles,

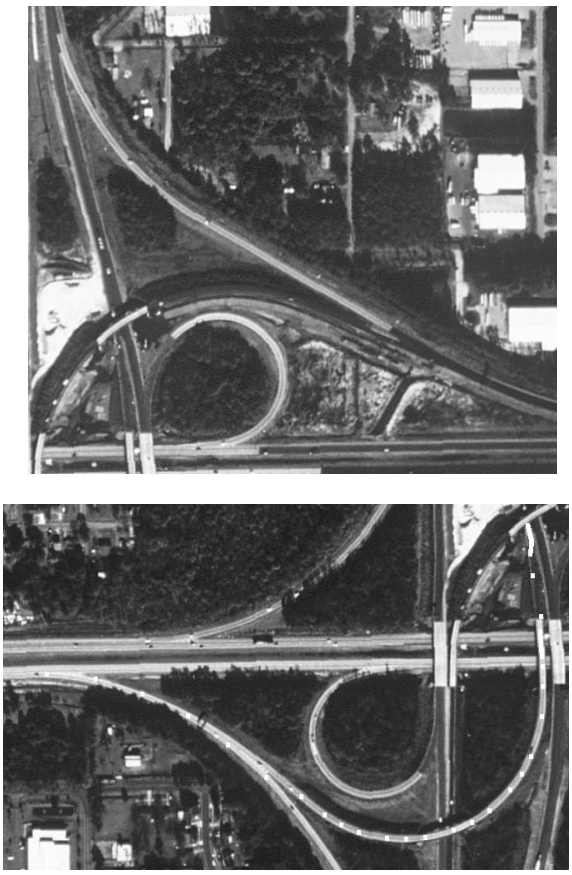

Figure 3: Top: A close-up view of an orthorectified aerial photograph of roads. Notice that the occlusion, crossing, and the change in road surface materials lead to abrupt change in road appearance. Bottom: An example of interactive road tracking. See text form details.

and the location and direction of the next example is decided by a weighted average of these profiles, where the weights are proportional to their distance from the separating hyperplane. In cases where too few on-road profiles exist, or where a good portion of the candidate profiles is within the ϵ-ball, the session ends and further input is requested from the user. Figure 3 bottom shows an example consisting of two sessions, starting from the top-right corner. White line segments indicate the locations of human inputs; white dots are the detected road axis points. When the road changes from dark to light, a human input is required to guide the tracker because the light road has not been experienced before.

We adopted the dataset from Zhou et. al. (2007). Our goal was to semi-automatically identify the 28 roads in one photograph of 14576×12749 pixels and 1m per pixel resolution of the Marietta area in Florida. Eight human participants were involved in this experiment. Each participant was asked to provide manual annotations of the road cen-

ter axes, and we simulated the semi-automatic process using the recorded human data as virtual users. Over the eight participants, accuracy results were similar (0.97 for both methods, corresponding to 1.95 vs. 1.85 pixels deviation from the ground-truth road center axes, for the proposed method and NNMA, respectively), but the proposed approach was able to work with much fewer human interactions (efficiency score 0.70 vs. 0.64 for the proposed method and NNMA, respectively).

Conclusion

In this paper, we presented a novel approach to sequential prediction and change detection, which often arise in interactive applications. This is achieved by devising an online-learning algorithm that naturally unifies the problems of prediction and change detection into a single framework. As a proof of concept, we applied the proposed approach to the interactive classification of a synthetic time-series, and we also experimented with a real-world road tracking task, where the proposed approach is shown to be competitive. For future work, we are looking into other interactive applications to apply the proposed framework.

Acknowledgments

NICTA is funded by the Australian Government as represented by the Department of Broadband, Communications and the Digital Economy and the Australian Research Council through the ICT Centre of Excellence program. This work is supported by the IST Program of the European Community, under the Pascal Network of Excellence, IST-2002-506778. WFB was supported by the Natural Sciences and Engineering Research Council of Canada.

References

Basseville, M., and Nikiforov, I. V. 1993. *Detection of abrupt changes: theory and application*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc.

Cheng, L.; Vishwanathan, S. V. N.; Schuurmans, D.; Wang, S.; and Caelli, T. 2006. Implicit online learning with kernels. In Schölkopf, B.; Platt, J.; and Hofmann, T., eds., *Advances in Neural Information Processing Systems 19*. Cambridge MA: MIT Press.

Chu, W., and Keerthi, S. 2005. New approaches to support vector ordinal regression. In *Proc. Intl. Conf. Machine Learning*.

Dietterich, T. G. 2000. Ensemble methods in machine learning. In *MCS '00: Proceedings of the First International Workshop on Multiple Classifier Systems*, 1–15. London, UK: Springer-Verlag.

Geman, D., and Jedynak, B. 1996. An active testing model for tracking roads in satellite images. *IEEE Trans. Pattern Anal. Mach. Intell.* 18(1):1–14.

Ho, S.-S. 2005. A martingale framework for concept change detection in time-varying data streams. In *ICML '05: Proceedings of the 22nd international conference on Machine learning*, 321–327. New York, NY, USA: ACM.

Hu, J.; Razdan, A.; Femiani, J.; Cui, M.; and Wonka, P. 2007. Road network extraction and intersection detection from aerial images by tracking road footprints. *IEEE Transactions on Geoscience and Remote Sensing* 45(12):4144–4157.

Kifer, D.; Ben-David, S.; and Gehrke, J. 2004. Detecting change in data streams. In *vldb'2004: Proceedings of the Thirtieth international conference on Very large data bases*, 180–191. VLDB Endowment.

Kivinen, J.; Smola, A. J.; and Williamson, R. C. 2004. Online learning with kernels. *IEEE Transactions on Signal Processing* 52(8).

Lacoste, C.; Descombes, X.; and Zerubia, J. 2005. Point processes for unsupervised line network extraction in remote sensing. *IEEE Trans. Pattern Anal. Mach. Intell.* 27(10):1568–1579.

Littlestone, N., and Warmuth, M. K. 1994. The weighted majority algorithm. *Inf. Comput.* 108(2):212–261.

Littlestone, N. 1988. Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. *Mach. Learn.* 2(4):285–318.

Merlet, N., and Zerubia, J. 1996. New prospects in line detection by dynamic-programming. *IEEE Trans. Pattern Anal. Mach. Intell.* 18(4):426–431.

Resnick, P., and Varian, H. R. 1997. Recommender systems. *Commun. ACM* 40(3):56–58.

Rochery, M.; Jermyn, I.; and Zerubia, J. 2006. Higher order active contours. *Intl. Journal of Computer Vision* 69(1):27–42.

Schölkopf, B., and Smola, A. 2002. *Learning with Kernels*. Cambridge, MA: MIT Press.

Vapnik, V. 1998. *Statistical Learning Theory*. New York: John Wiley and Sons.

Vovk, V. G. 1995. A game of prediction with expert advice. In *COLT '95: Proceedings of the eighth annual conference on Computational learning theory*, 51–60. New York, NY, USA: ACM.

Wang, J.; Bhat, P.; Colburn, R. A.; Agrawala, M.; and Cohen, M. F. 2005. Interactive video cutout. In *SIGGRAPH '05: ACM SIGGRAPH 2005 Papers*, 585–594. New York, NY, USA: ACM.

Yuille, A. L., and Coughlan, J. M. 2000. Fundamental limits of bayesian inference: Order parameters and phase transitions for road tracking. *IEEE Trans. Pattern Anal. Mach. Intell.* 22(2):160–173.

Zhou, J.; Cheng, L.; and Bischof, W. 2007. Online learning with novelty detection in human-guided road tracking. *IEEE Transactions on Geoscience and Remote Sensing* 45(12):3967–3977.