Data Stream Prediction Using Incremental Hidden Markov Models

Kei Wakabayashi and Takao Miura

HOSEI University, Dept.of Elect.& Elect. Engr.,
3-7-2 KajinoCho, Koganei, Tokyo, 184–8584 Japan

Abstract. In this paper, we propose a new technique for time-series prediction. Here we assume that time-series data occur depending on *event* which is unobserved directly, and we estimate future data as output from the most likely event which will happen at the time. In this investigation we model time-series based on event sequence by using *Hidden Markov Model*(HMM), and extract time-series patterns as trained HMM parameters. However, we can't apply HMM approach to *data stream* prediction in a straightforward manner. This is because Baum-Welch algorithm, which is traditional unsupervised HMM training algorithm, requires many stored historical data and scan it many times. Here we apply *incremental Baum-Welch algorithm* which is an on-line HMM training method, and estimate HMM parameters dynamically to adapt new time-series patterns. And we show some experimental results to see the validity of our method.

Keywords: Forecasting, Data Stream, Hidden Markov Model, Incremental Learning.

1 Introduction

Recently there have been a lot of knowledge-based approaches for huge databases, and much attention have been paid on *time-series prediction* techniques [2]. Especially prediction on *data stream*, which is assumed huge amount of data and high speed updating, is important technique in many domains.

There have been many prediction approach proposed so far [4]. One of the traditional approach is *Exponential Smoothing* that is a heuristic method based on weighted mean of past data. In exponential smoothing method, we assume weight of data decrease with time exponentially. Then we can obtain the weighted mean at time t using the weighted mean at time $t-1$ as an aggregated value.

Holt-Winters method is one of the famous prediction approach proposed based on exponential smoothing [3]. In this method they estimate future data based on the weighed mean $\tilde{y}_t$ and mean of $\tilde{y}_t$'s variation F_t. $\tilde{y}_t$ and F_t are updated at each time as follows:

$$\tilde{y}_t = \lambda_1 y_t + (1-\lambda_1)(\tilde{y}_{t-1} + F_{t-1})$$

$$F_t = \lambda_2(\tilde{y}_t - \tilde{y}_{t-1}) + (1-\lambda_2)F_{t-1}$$

T.B. Pedersen, M.K. Mohania, and A M. Tjoa (Eds.): DaWaK 2009, LNCS 5691, pp. 63–74, 2009.

λ_1 and λ_2 are called as smoothing parameters. Here $0.0 \leq \lambda \leq 1.0$. If λ is set to bigger value, the weight of past data will decrease more quickly. They estimate future data at time $t + h$ using $\tilde{y}_t$ and F_t as follows:

$$\tilde{y}_{t+h|t} = \tilde{y}_t + hF_t$$

However, by Holt-Winters method it is hard to estimate data depending on event. For example, wind velocity data occur depending on event such as approach of typhoon, does not depend on previous data directly.

In Hassan [5], they discuss how to predict stock market using *Hidden Markov Model* (HMM) which is a stochastic model assumed simple Markov process with hidden state. They estimate hidden state at each observation and estimate future observations based on state transition. We believe we can estimate time-series depending on event sequence effectively by this approach, since the estimated data doesn't depend on previous observation.

However this approach can't estimate a data following new pattern which didn't be appeared in training data, because they train HMM parameters in advance by EM-algorithm which requires large computational time. This is serious problem especially in *data stream* forecasting. Data stream is assumed as infinite time-series and distribution of data changes dynamically [6] [10]. We should train the model incrementally while forecasting data since given training data contain only a part of patterns in whole data stream.

There are some research about incremental learning for HMM [1]. In Stenger [11], they discuss how to update HMM parameters incrementally and propose *incremental Baum-Welch algorithm*. Incremental Baum-Welch algorithm does not require historical data, and we can compute very quickly. They apply this approach to background modeling in real-time video processing.

In this investigation we propose a new forecasting approach using incremental Baum-Welch algorithm. We believe we can achieve adaptive time-series estimation on data stream by using incremental HMM learning. We compare our approach to conventional batch Baum-Welch algorithm and show the effectiveness of proposal method.

In this work, we discuss time-series forecasting issue on data stream in section 2. In section 3 we review incremental Hidden Markov Model and we develop the forecasting technique using incremental HMM. Section 4 contains experimental results. We conclude our discussion in section 5.

2 Time-Series Forecasting Based on Event

First of all, we describe how we predict time-series on data stream. We illustrate our approach in a figure 1. Let us assume a sales history of goods in a store as time-series, such that has a large sale for coffees at day t, beer at day $t + 1$, and fruit juice at day $t + 2$.

In this investigation, we assume that observation in time-series occurs depending on unknown (hidden) state. In the figure 1, we interpret the day t as a busy day for many customer by examining the sales data, because of few beers and

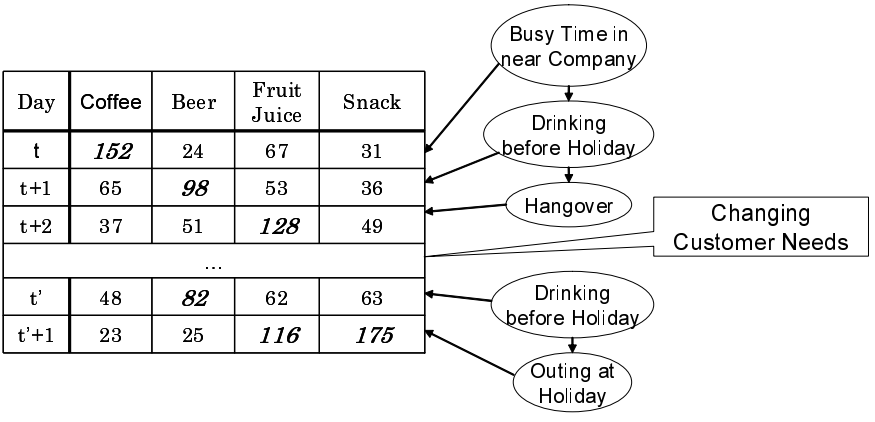

Fig. 1. Modeling Time-Series based on Event Sequence

many coffees saled. We consider the day $t + 1$ as a day before holiday because many customers want to drink alcohol, and many customers have hangover at day $t + 2$ because of many fruit juices sold.

On this interpretations of data, we see transition from a busy day to a drinking day, and transition from a drinking day to a hangover day. If we see these transition pattern in past data many times, we may estimate sales of goods at $t + 1$ or $t + 2$ when we are at the day t.

In this approach, although we estimate future observation by considering transition of hidden state, we don't have to interpret these unknown state [12]. In this investigation we formalize hidden state explicitly using Hidden Markov Model (HMM), and estimate observation by HMM trained using past observations without interpreting states.

However, during the passage of long time, some change of transition pattern may occur. For example, if a new high school opens near by the store, many students will come to the store and sales of snacks and juices will increase. For such changes of pattern, we should consider adaptation to new pattern even while processing estimation of future observations.

Unfortunately, it is not easy to update HMM parameters *on-line*, that means training model using new observations dynamically after start of process for estimation. This is because we can't obtain the most likely parameters directly from observation sequence, but we must estimate hidden state sequence before parameter estimation. Generally, for unsupervised learning of HMM, we employ Baum-Welch algorithm which requires training observation sequence and scans them many times. However, it is hard to recalculate HMM parameters by Baum-Welch algorithm at each time we get new observation, from the aspect of computational time.

Especially in data stream environment, we can't memorize historical observation sequence because of infinite amount of observation, then we should apply *incremental* parameter updating. In this investigation, we employ *incremental*

Baum-Welch algorithm which doesn't require historical observations for on-line update of HMM parameters. We apply incremental Baum-Welch algorithm to estimating future observation and we discuss how we can achieve adaptive forecasting with on-line training.

3 Forecasting Using Incremental Hidden Markov Model

3.1 Hidden Markov Model

A *Hidden Markov Model* (HMM) is nothing but an automaton with output where both the state transition and the output are defined in a probabilistic manner. The state transition arises according to a simple Markov model but it is assumed that we don't know on which state we are standing now[1], and that we can observe an output symbol at each state. We could estimate the transition sequences through observing output sequence.

A HMM model consists of (Q, Σ, A, B, π) defined below [8]:

(1) $Q = \{q_1, \cdots, q_N\}$ is a finite set of states
(2) $\Sigma = \{o_1, \cdots, o_M\}$ is a set of output symbols.
(3) $A = \{a_{ij}, i, j = 1, ..., N\}$ is a probability matrix of state transition where each a_{ij} means a probability of the transition at q_i to q_j. Note $a_{i1}+...+a_{iN} = 1.0$.
(4) $B = \{b_i(o_t), i = 1, ..., N, t = 1, ..., M\}$ is a probability of outputs where $b_i(o_t)$ means a probability of an output o_t at a state q_i.
(5) $\pi = \{\pi_i\}$ is an initial probability where π_i means a probability of the initial state q_i. In this work, we assume all states occur in the same probability as initial state, that is $\pi_i = \frac{1}{N}$.

The probability matrix A shows the transition probability within a framework of simple Markov model, which means state change arises in a probabilistic manner depending only on the current state. Thus, for instance, the (i, j)-th component of A^2 describes the transition probability from q_i to q_j with two *hops* of transitions. Similarly the output appears depending only on the current state.

Since we consider multidimensional observation, we define output symbol as numeric vector in this work. We assume each values in vector occurs from normal distribution independently. That is, when $o_t = o_{t1}, o_{t2}, ..., o_{tD}$, we define probability of o_t at state q_i as:

$$b_i(o_{tk}) = \frac{1}{\sqrt{2\pi}\sigma_{ik}} e^{-\frac{(o_{tk}-\mu_{ik})^2}{2\sigma_{ik}^2}}$$

μ_{ik} and σ_{ik} are the mean and variance of normal distribution respectively, and both are parameters of our HMM.

[1] This is why we say *hidden*.

We should think about how to obtain parameters, since it is hard to determine the transition probability matrix A and the output probability B definitely. This problem is called a *model calculation* of HMM. Usually we do that by means of some machine learning techniques[9].

One of the typical approach is *supervised learning*. In this approach, we assume *labeled training data* in advance to calculate the model, but the data should be correctly classified by hands since we should extract typical patterns them by examining them. Another approach comes, called *unsupervised learning*. Assume we can't get labeled training data but a mountain of unclassified data except a few. Once we obtain strong similarity between the classified data and unclassified data (such as high correlation), we could extend the training data in a framework of Expectation Maximization (EM) approach[7].

One of the typical approach is known as a *Baum-Welch* algorithm. The algorithm has been proposed based on EM approach. That is, the algorithm adjusts the parameters many times to maximize the likelihood for the generation of the output symbols given as unsupervised data. The process goes just same as EM calculation, i.e., we calculate the expect value of the transition probability and the output probability, then we maximize them. We do that until few change happens.

3.2 Incremental Baum-Welch Algorithm

Let $\gamma_t(q_i)$ be the stay probability on state q_i at time t, and $R_T(q_i)$ be the summation of stay probabilities $\gamma_t(q_i)$, that is:

$$R_T(q_i) = \sum_{t=1}^{T} \gamma_t(q_i) \tag{1}$$

When we obtain a new observation o_T at time T, we update HMM parameters by using *incremental Baum-Welch algorithm*. In first, we calculate $\gamma_T(q_x)$ which is likelihood that we stay state q_x at time T. Here we assume $\gamma_{T-1}(q_i)$ and $R_{T-1}(q_i)$ are memorized in advance for all i. Then, we obtain $\gamma_T(q_i, q_x)$ which is the probability of the transition at time T from state q_i to q_x using current HMM parameters as:

$$\gamma_T(q_i, q_x) = \gamma_{T-1}(q_i) \frac{a_{ix} b_x(o_T)}{\sum_{x=1}^{N} a_{ix} b_x(o_T)}$$

We obtain $\gamma_T(q_x)$ using $\gamma_T(q_i, q_x)$ as:

$$\gamma_T(q_x) = \sum_{i=1}^{N} \gamma_T(q_i, q_x)$$

Then the HMM parameters are updated as [11]:

$$a'_{ij} = \frac{R_{T-1}(q_i) a_{ij} + \gamma_T(q_i, q_j)}{R_T(q_i)}$$

$$\mu_i' = \frac{\sum_{t=1}^{T} \gamma_t(q_i) o_t}{\sum_{t=1}^{T} \gamma_t(q_i)} = \frac{R_{T-1}(q_i)\mu_i + \gamma_T(q_i) o_T}{R_T(q_i)}$$

$$\sigma_i' = \frac{\sum_{t=1}^{T} \gamma_t(q_i)(o_t - \mu_i)^2}{\sum_{t=1}^{T} \gamma_t(q_i)} = \frac{R_{T-1}(q_i)\sigma_i + \gamma_T(q_i)(o_T - \mu_i)^2}{R_T(q_i)}$$

By using incremental Baum-Welch algorithm, we can update parameters quickly without historical data.

However, change of parameters become smaller and smaller with time because R_T increase directly with time and the weight of new observation become relatively small. Therefore we define *forget function* $w_\lambda(t)$ to reduce the weight of old observations as:

$$w_\lambda(t) = \lambda^t$$

Here let t be the time elapsed since the data observed, λ be constant of forgetting speed and $0.0 \leq \lambda \leq 1.0$. In this paper, we call $\lambda' = (1.0 - \lambda)$ as *forget coefficient*.

We redefine $R_T(q_i)$ to reduce the weight of old observations as below:

$$\begin{aligned} R_T(q_i) &= \sum_{t=1}^{T} \gamma_t(q_i) \lambda^{T-t} \\ &= \sum_{t=1}^{T-1} \gamma_t(q_i) \lambda^{T-1-t} \lambda + \gamma_T(q_i) \lambda^{T-T} \\ &= \lambda R_{T-1}(q_i) + \gamma_T(q_i) \end{aligned}$$

We can derive $R_T(q_i)$ recursively. By applying this new definition, we can update HMM parameters using observations weighted based on forget function by incremental Baum-Welch algorithm.

3.3 Estimating Observation

We develop our theory to estimate future data using incremental Hidden Markov Model. Since incremental Baum-Welch algorithm does not execute iterative parameter estimation such as EM-algorithm, it requires many observations for convergence HMM parameters initialized random values. Here we apply Baum-Welch algorithm as off-line learning to obtain initial model.

First, we generate random numbers for HMM parameters A and μ, and assign 1.0 to all σ. Then we apply Baum-Welch algorithm to given training sequence which contains T observations. After convergence, we calculate $\gamma_t(q_i)$, the stay probabilities on state q_i at time t for $1 \leq t \leq T$. We obtain its summation $\sum_{t=1}^{T} \gamma_t(q_i)$ and the stay probabilities at final time T, $\gamma_T(q_i)$.

Then we discuss how to estimate the observation at next time $T+1$ using HMM. Since observation at time $T+1$ is unknown, we estimate the likelihood which we stay on state q_i at time $T+1$ as:

$$\gamma_{T+1}(q_i) = \sum_{j=1}^{N} \gamma_T(q_j) a_{ji}$$

If we are standing at state q_i, the most likely observation at time $T+1$ is $\mu_i = (\mu_{i1}, \mu_{i2}, ..., \mu_{iD})$. Therefore, the expected value of the observation is estimated as:

$$\sum_{j=1}^{N} \gamma_{T+1}(q_i)\mu_i$$

We generate this value as our estimated observation at time $T+1$.

When we observe a new data at time $T+1$, we update HMM parameters by using incremental Baum-Welch algorithm. After updating parameters, we calculate $\gamma_{T+1}(q_i)$ and $R_{T+1}(q_i)$ on each state q_i at time $T+1$ using updated HMM parameters. That is:

$$\gamma_{T+1}(q_i) = \sum_{j=1}^{N} \gamma_T(q_j) a'_{ji} b'_i(o_{T+1})$$
$$R_{T+1}(q_i) = \lambda R_T(q_i) + \gamma_{T+1}(q_i)$$

We use $\gamma_{T+1}(q_i)$ to estimate next observation.

4 Experimental Results

Here we discuss some experimental results to show the usefulness of our approach. First we show how to obtain the results and to evaluate them, then we show and examine the results.

4.1 Preliminaries

As a test data for our experiments, we take 5 years (1828 days) data in Himawari Weather Data 1996 to 2000 in Tokyo, and select 6 schema, *maximum temperature* (℃), *minimum temperature* (℃), *humidity* (%), *maximum wind velocity* (m/s), *duration of sunshine* (hour) and *day rainfall amount* (mm). Table 1 show a part of observation sequence.

We take 10% of observation sequence (182 days) for off-line training data, and we predict the last 90% of observations (1646 days). Here we apply 2 algorithms to estimate observations, HMM method with batch training by conventional Baum-Welch algorithm periodically (batch HMM), and HMM method with incremental training by incremental Baum-Welch algorithm at every observation(Incremental HMM). In batch HMM method, we memorize data observed

Table 1. Test Data

Day	Max. Temp.	Min. Temp.	Humidity	Max. Wind	Sunshine	Rainfall
3/26/1997	17.5	10.4	49	11.3	9.4	0.0
3/27/1997	11.9	8.4	84	16.1	0.1	14.5
3/28/1997	17.2	6.3	47	14.4	9.6	0.0
3/29/1997	17.4	12.0	64	16.9	0.0	10.5
3/30/1997	25.1	10.4	63	17.8	9.9	40.0
3/31/1997	14.6	7.9	47	10.6	7.5	0.0
4/1/1997	15.7	8.0	50	11.5	8.4	0.0

at past w_1 days, and update parameters at every w_2 days. In this experiment we set w_1 (window size) to $100, 200, 400, 800, \infty$, and w_2 (interval) to $1, 10, 20, 40$, and we examine results. In incremental HMM method, we set λ' (forget coefficient) to $0.0, 0.001, 0.002, 0.005, 0.01, 0.02, 0.05, 0.1, 0.2, 0.5$, and we examine results.

We evaluate the prediction accuracy of method using *Mean Square Error* (MSE). Let p_t be the estimated value, x_t be the actual value at time t, then MSE is defined as follows:

$$MSE = \frac{1}{T}\sum_{t=1}^{T}(p_t - x_t)^2$$

We also examine execution time of estimating observation sequence (1646 days) by batch HMM and incremental HMM method. We evaluate MSE and execution time by average of 10 trials. In the following experiments, we give 15 HMM states.

4.2 Results

Table 2 shows the MSE results. In result of maximum temperature and minimum temperature, we got the best MSE 10.17, 5.01 by incremental HMM method in condition of $\lambda' = 0.2, 0.5$, respectively. The best MSE of maximum and minimum temperature prediction by batch HMM method is 18.21, 14.93 respectively, both are given by condition of $w_1 = 200$ and $w_2 = 10$. From this result, we can say that new observations are more important to forecast future temperature data, because the weight of old observations are reduced quickly by setting λ' to 0.2, 0.5 or w_1 to 200. We can see same characteristic in humidity results, that we got the best MSE by incremental HMM method in condition of $\lambda' = 0.2$, and in batch HMM method we got the best MSE in condition of $w_1 = 200$ and $w_2 = 10$.

On the other hand, in result of rainfall amount, we see that better result comes from conditions of lower forget coefficient, larger window size, and small interval of update. We got the best MSE of rainfall prediction, 175.17, by incremental HMM method in condition of $\lambda' = 0.01$. In batch HMM result we got the best MSE, 177.01, in condition of $w_1 = \infty$ and $w_2 = 10$. From this result we can say that both new and old observations are important to forecast future data,

Table 2. Prediction Accuracy (MSE)

Method		MSE					
		Max. Temp.	Min. Temp.	Humidity	Max. Wind	Sunshine	Rainfall
Batch HMM							
window	interval						
100	1	22.42	18.96	191.46	19.16	16.79	186.45
100	10	25.64	22.75	194.54	19.06	16.54	185.43
100	20	23.85	20.87	191.25	19.20	16.61	186.29
100	40	24.31	21.06	195.50	19.23	16.67	186.64
200	1	19.68	17.08	177.19	19.03	15.87	177.34
200	10	**18.21**	**14.93**	**169.79**	18.92	15.62	177.71
200	20	28.87	27.63	185.51	18.86	15.66	180.10
200	40	28.79	26.35	181.91	18.94	15.66	181.16
400	1	33.16	31.83	204.17	**18.81**	15.67	176.75
400	10	25.55	23.21	188.69	18.90	15.70	177.64
400	20	32.87	32.15	196.62	19.14	15.71	179.52
400	40	24.60	22.09	179.01	19.12	15.65	181.05
800	1	57.52	59.54	244.36	18.88	15.87	177.64
800	10	20.80	17.82	179.98	18.94	15.63	177.59
800	20	29.62	27.90	187.00	19.01	**15.60**	180.15
800	40	30.98	29.33	188.29	19.05	15.68	180.81
∞	1	56.68	58.34	246.79	18.94	15.81	177.63
∞	10	20.29	17.32	178.78	18.91	15.61	**177.01**
∞	20	31.82	30.25	190.85	18.94	15.62	178.95
∞	40	31.97	30.36	188.47	18.96	15.67	180.63
Inc. HMM							
λ'							
0.0		59.93	61.58	236.72	18.91	15.62	175.25
0.001		57.35	58.69	232.39	18.86	15.60	175.99
0.002		58.15	59.76	233.40	18.92	15.58	176.21
0.005		53.94	54.99	225.99	18.79	15.40	175.50
0.01		48.09	47.89	217.26	19.01	15.34	**175.17**
0.02		40.58	34.79	255.31	21.10	15.82	176.73
0.05		15.97	12.34	166.98	**18.29**	**15.33**	179.76
0.1		11.88	7.69	158.14	18.50	15.35	183.70
0.2		**10.17**	5.61	**151.93**	19.04	15.64	188.39
0.5		10.52	**5.01**	167.52	22.19	17.80	220.41

because frequent parameter update using many past observations cause better MSE.

Table 3 shows the execution time of estimating observations. In batch HMM method, the execution time increase directly with window size and inversely with interval of update. By incremental HMM method we can predict very quickly compared to batch HMM method. For instance, compared to batch HMM method in condition of $w_1 = 800$ and $w_2 = 1$ that takes 3568244.20ms, we can execute incremental HMM method for about 1/6500 of time.

Table 3. Execution Time

Method		Execution Time(ms)	
		Whole Data	per 1 Data
Batch HMM			
window	interval		
100	1	537782.10	326.72
100	10	53922.80	32.76
100	20	27141.00	16.49
100	40	13535.20	8.22
200	1	1060462.50	644.27
200	10	105598.40	64.15
200	20	52573.70	31.94
200	40	26112.40	15.86
400	1	2042269.70	1240.75
400	10	139620.30	84.82
400	20	59092.60	35.90
400	40	27025.40	16.42
800	1	3568244.20	2167.83
800	10	139652.60	84.84
800	20	58978.80	35.83
800	40	27006.90	16.41
∞	1	5342431.90	3245.71
∞	10	141042.70	85.69
∞	20	59390.80	36.08
∞	40	26963.00	16.38
Inc. HMM			
λ'			
0.0		550.10	0.33
0.001		537.60	0.33
0.002		551.90	0.34
0.005		545.40	0.33
0.01		542.10	0.33
0.02		561.00	0.34
0.05		545.40	0.33
0.1		543.60	0.33
0.2		542.20	0.33
0.5		545.60	0.33

4.3 Discussions

Here we discuss how we can think about our experimental results and especially about our approach.

In result of maximum temperature, minimum temperature and humidity, we got better MSE when we reduce the weight of old observations quickly. This is because these attribute have strong characteristic of *seasonality*, for instance, in summer period it is around 30 ℃ of temperature and 70 % of humidity at most day, and in winter period 10 ℃ of temperature and 35 % of humidity. Because

season changes with long period, we can construct specific model for each seasons by reducing weight of old observations quickly, and get good prediction accuracy.

On the other hand, in rainfall amount prediction, we got better MSE by using many past observations. This is because we have to learn the weather patterns based on observation vector sequence to forecast rainfall amount. In other words, rainfall amount strongly depends on weather event (like sunny, rain or typhoon). Frequent update of parameters is also nessesary to forecast rainfall because small w_2 cause better MSE.

In the result of execution time, incremental HMM method requires few execution time compared to batch HMM method. Especially, rainfall amount prediction requires large window size and small update interval, therefore it takes a lot of time for good forecasting by batch HMM method. By using proposal method, we can forecast observations very quickly with about the same accuracy as batch HMM method. In addition, proposal method doesn't require large storage to keep old observations, therefore any storage access doesn't happen even if data amount become huge.

5 Conclusion

In this investigation we have proposed how to estimate future data on data stream by modeling time-series as stochastic process. We have presented adaptive future data estimation using HMM by applying incremental Baum-Welch algorithm. We have discussed some experimental results and shown the usefulness of our approach.

In this work we have given the number of state to HMM in advance. But we need some techniques which can vary number of state concurrently for adapting new patterns more accurately and quickly.

In this case, we have examined numeric vector as observations, but it is possible to apply our approach to nonnumeric time-series. For example, we may have some sort of forecast for accidents by applying our approach to news stream.

References

1. Cavalin, P.R., Sabourin, R., Suen, C.Y., Britto Jr., A.S.: Evaluation of Incremental Learning Algorithms for An HMM-Based Handwritten Isolated Digits Recognizer. In: The 11th International Conference on Frontiers in Handwriting Recognition (ICFHR 2008), Montreal, August 19-21 (2008)
2. Duan, S., Babu, S.: Processing forecasting queries. In: Proc. of the 2007 Intl. Conf. on Very Large Data Bases (2007)
3. Gelper, S., Fried, R., Croux, C.: Robust Forecasting with Exponential and Holt-Winters Smoothing (June 2007)
4. de Gooijer, J.G., Hyndman, R.J.: 25 Years of IIF Time Series Forecasting: A Selective Review, Tinbergen Institute Discussion Papers 05-068/4 (2005)
5. Md. Hassan, R., Nath, B.: StockMarket Forecasting Using Hidden Markov Model: A New Approach. In: Proceedings of the 5th International Conference on Intelligent Systems Design and Applications (2005)

6. Jian, N., Gruenwald, L.: Research Issues in Data Stream Association Rule Mining, SIGMOD Record 35-1, pp.14–19 (2006)
7. Iwasaki, M.: Statistic Analysis for Incomplete Data. EconomistSha, Inc. (2002) (in Japanese)
8. Manning, C.D., Schutze, H.: Foundations of Statistical Natural Language Processing. MIT Press, Cambridge (1999)
9. Mitchell, T.: Machine Learning. McGrawHill Companies (1997)
10. Muthukrishnan, S.: Data streams: algorithms and applications. In: Proceedings of the fourteenth annual ACM-SIAM symposium on discrete algorithms (2003)
11. Stenger, B., Ramesh, V., Paragios, N.: Topology free Hidden Markov Models: Application to background modeling. In: IEEE International Conference on Computer Vision (2001)
12. Wakabayashi, K., Miura, T.: Identifying Event Sequences using Hidden Markov Model. In: Kedad, Z., Lammari, N., Métais, E., Meziane, F., Rezgui, Y. (eds.) NLDB 2007. LNCS, vol. 4592, pp. 84–95. Springer, Heidelberg (2007)