Learning and Sequential Decision Making For Medical Data Streams Using RL Algorithm

Prof. Pramod Patil
College of Engineering
Pune-18

Dr. Parag Kulkarni
College of Engineering
Pune-18

Ms. Rachana Shirsath
Dr. D.Y. Patil Institute of Engineering and Technology, Pimpri, Pune-18

Abstract

Data stream mining has obtained a high attraction due to the importance of its applications and increase in the generation of streaming information. The data streams are the set of data, which are moving in specified distribution. Extracting knowledge structures represented in models and patterns in non stopping streams of information is known as data stream mining. Increasing access to incredibly large, non stationary data sets and corresponding demands to analyze these data has led to use reinforcement learning algorithm on data streams. For diabetes data set it is difficult to differentiate between the value of blood glucose level and value of insulin doses. Also it is difficult to take decision about giving specific quantity of insulin dose to the diabetes patient.

In this paper we implement the reinforcement learning algorithm on diabetes data streams. The algorithm process the data streams and differentiate the values for blood glucose level and insulin dose and take the decision for next insulin dose. Depending on state and action taken the payoff is assign to the decision. This helps in classifying the data for diabetes doses and also helps in making decisions for giving specific quantity of dose at a particular time. In comparison with other methods our proposed algorithm is faster. A proposed methodology is tested on diabetes data.

Key words: data stream, concept drift, Reinforcement learning, RL Algorithms.

1. Introduction

Data streams pose several unique problems that inhibit the application of standard data mining techniques. The dataset is continuously online and growing to include new measurements; therefore, effective algorithms for analyzing these data must be able to work within a constant memory footprint. In particular, the entire dataset cannot be stored in memory and historical data must eventually be forgotten. An additional problem is that the probability distributions associated with the data might change over time, a condition known as concept drift. Any reasonable stream learning algorithm must be able to recognize and cope with these situations.

One of the most important research fields in data stream mining [1] community by building prediction models from data streams. Recently, many ensemble models have been proposed to build prediction models from concept drifting data streams. Different from traditional incremental and online learning approaches that merely rely on a single model, ensemble learning employs a divide-and-conquer approach to first split the continuous data streams into small data chunks, and then build light-weight base classifiers from the small chunks. At the final stage, all base classifiers are combined together for prediction. In corresponding, new methods are needed for extracting knowledge from fast-moving, quickly changing, and extremely large data sources for increasing data storage capacity and processing power. The growing field of data stream mining addresses these problems.

The dataset is continuously online and growing to include new measurements; therefore, effective algorithms for analyzing these data must be able to work within a constant memory footprint. In particular, the entire dataset cannot be stored in memory and historical data must eventually be forgotten. Data stream mining systems must also cope with missing and corrupted data: noisy communication lines, human error, experimental design, and failing sensors can all alter and interrupt data streams. In online classification systems, both observations and labels can be missing or corrupted at any time. Noisy and missing observations have been the subject of extensive research. Observation noise is often explicitly modelled by learning procedures, and various imputation techniques have been proposed for handling missing values.
Learning is usually formulated as a search conducted in an abstractly defined space, and a large collection of understanding and designing procedures, or algorithms, for enabling a device or program to improve its performance over time. The behavior observed in classical conditioning experiments is far from computationally trivial; its strongest ties are to mathematical theories and computational procedures that are exceedingly useful in practice and surprisingly complex. TD Model as a conditional method that can be useful in solving engineering problems. TD methods can be used as components of synthetic learning systems. TD Procedure is related to theoretical principals who serve both to explain the operations of TD methods and to connect them to existing theories of prediction, control and learning.

2. Literature Survey

Gaber et al. [1] review the theoretical foundations of data stream analysis. Mining data stream systems, techniques are critically reviewed. Finally, they outline and discuss research problems in streaming mining field of study. These research issues should be addressed in order to realize robust systems that are capable of fulfilling the needs of data stream mining applications.

Garnett et al [2] review about data streams which pose several unique problems that inhibit the application of standard data mining techniques. He also review that dataset is continuously online and growing to include new measurements; therefore, effective algorithms for analyzing these data must be able to work within a constant memory footprint. In particular, the entire dataset cannot be stored in memory and historical data must eventually be forgotten. An additional problem is that the probability distributions associated with the data might change over time, a condition known as concept drift. Any reasonable stream learning algorithm must be able to recognize and cope with these situations.

Leslie Pack Kaelbling et al [3] have discussed issues of reinforcement learning, which include trading off exploration and exploitation. Also provide the foundations of field using Markov decision theory. learning from delayed reinforcement, constructing empirical models to accelerate learning, making use of generalization and hierarchy, and coping with hidden state. Also they have given survey of some implemented systems and an assessment of the practical utility of current methods for reinforcement learning.

Marco A. Wiering and Hado van Hasselt [4] have present several ensemble methods used for combining multiple different reinforcement learning (RL) algorithms by using single agent. Learning speed and final performance get enhanced by combining the chosen actions or action probabilities of different RL algorithms. They have designed and implemented four different ensemble methods combining the following five different RL algorithms: Q learning, Sarsa, actor–critic (AC), QV - learning, and AC learning automaton.

M. Mill´an-Giraldo et al [6] discussed about some applications in which data arrive sequentially and they are not available in batch form, what makes difficult the use of traditional classification systems. In addition, some attributes may lack due to some real-world conditions. For this problem, a number of decisions have to be made regarding how to proceed with the incomplete and unlabeled incoming objects, how to guess its missing attributes values, how to classify it, whether to include it in the training set, or when to ask for the class label to an expert. Unfortunately, no decision works well for all data sets. This data dependency motivates our formulation of the problem in terms of elements of reinforcement learning. The application of this learning paradigm for this problem is, to the best of our knowledge.

Gregory Ditzler et al [11] discussed about learning in non-stationary environments, also known as learning concept drift, is concerned with learning from data whose statistical characteristics change over time. Concept drift is further complicated if the dataset is class-imbalanced. They have addressed two issues independently, and their joint treatment has been mostly underexplored. They describe two ensemble-based approaches for learning concept drift from imbalanced data. First approach is a logical combination NSE algorithm for concept drift, with the well-established SMOTE for learning from imbalanced data. And second approach makes two major modifications to Learn++. NSE-SMOTE integration by replacing SMOTE with a sub ensemble that makes strategic use of minority class data; and replacing Learn++.NSE and its class-independent error weighting mechanism with a penalty constraint that forces the algorithm to balance accuracy on all classes.

3. Reinforcement Learning
Reinforcement learning is a way of programming agents by reward and punishment. This is a problem faced by an agent who learns behaviour by trial and error method in dynamic environment. There are basic two strategies for solving reinforcement learning problem. The first is to find one who performs well in given environment. This method works well in genetic programming and genetic algorithms. The second method is to use statistical techniques and dynamic programming methods for estimating the utility of taking actions in states of the world. [3]Reinforcement learning differs from standard supervised learning in that correct input/output pairs are never presented, nor sub-optimal actions explicitly corrected. An agent using reinforcement learning interacts with its environment by receiving states and rewards from the environment, and generating actions that change the environment.

\[ V_\pi(x) = E_\pi \left[ \sum_{t=1}^{\infty} \gamma^t r_t + 1 | x_0 = x \right] \]

The function is the evaluation function for policy \( \pi \). For each state \( x \), \( V_\pi(x) \) is the expected return over the infinite number of time steps beginning at \( t=0 \) under the conditions that the system begins in state \( x \) and the agent uses policy \( \pi \), where it is understood that the discount factor, \( \gamma \), has some specified value. We call \( V_\pi(x) \) the evaluation of state \( x \). For the route finding example, the evaluation of location \( x \) for policy \( \pi \) depends on the expected number of time steps that agents takes to reach the goal from location \( x \) using policy \( \pi \). If \( \pi \) always brings the agent to the goal in say, \( n \) time steps from a location \( x \), then.

Let \( \pi \) and \( \pi' \) be any two policies. Policy \( \pi' \) is an improvement over policy \( \pi \) if the expected return of \( \pi' \) is no smaller than that for \( \pi \) for any state, and is larger for at least one state. More precisely, \( \pi' \) is an improvement over \( \pi \) if \( V_\pi(x) >= V_\pi'(x) \) for all states \( x \), with strict inequality holding for at least one state. A policy is an optimal policy, which we denote \( \pi^* \), if no policy is an improvement over it. Because the optimality of policies depends on the discount factor, technically we should refer to \( \gamma \)-optimal policies. As \( \gamma \) changes, different policies become optimal because a policy best for short-term versions of a task will generally not be best for long-term versions. Whatever policies are compared they are compared according to expected returns defined for the same \( \gamma \). For any optimal policy \( \pi^* \),

\[ v_{\pi'}(x) \geq v_{\pi^*}(x) \]
For all states x, an agent using an optimal policy will maximize the expected return from any system state. The object of a sequential decision task is to find one of the optimal policies.

4. Proposed System

In Our Proposed System we take Insulin data base which contains the information of the blood glucose level of the patient at various instances. Blood glucose level of the patient before and after meal is mentioned. This database is analyzed and classified in to various states. If before meal blood glucose level is between 70 to 100 then it is classified as constant , if it is less than 70 then classify as down and if greater than 100 then classify as up. After meal blood glucose level is between 80 to 140. Depending on previous conditions of the patient the decision about the next dose is taken. System gives the output as how much quantity of insulin is given to the patient.

Fig [3]. Decision for Insulin Dose

If decision of dose is correct and at next measurement the blood glucose level is normal then reward is given to the decision else penalty is given. This mechanism of reward and penalty helps in improving the decision in next states.

Figure 3 shows the decision made by system. For example for regular insulin dose we have to give 11 unit of insulin to the patient.

Fig.[4] Variation of Blood Glucose level over time.

Blood glucose level of the diabetes patient is always changing over time. Figure 4 shows the variation of blood glucose level at different time intervals.

Conclusion

Data streams are considered as the continuous flow of data from a source to destination. Data Stream Mining is the process of extracting knowledge structures from continuous, rapid data records. Reinforcement learning promises a beguiling way of programming agents by reward and punishment without needing to specify how the task is to be achieved. The task improving a decision policy within the framework developed here is similar to the task faced by an animal. The amount of payoff received by agent is depending on its actions and feature vectors of the state. The method used by agent to adjust decision policy, evaluation function corresponding to a policy, such as TD procedure has obvious utility. Using reinforcement learning algorithm on diabetes data helps in making effective decision about giving specific quantity of insulin dose that should be given to the patient at particular time.

References


