© 2017 IJSRSET | Volume 3 | Issue 7 | Print ISSN: 2395-1990 | Online ISSN : 2394-4099
Themed Section: Engineering and Technology
National Level Technical Conference on Advanced Computing Technologies- n'CACT'17, Department of Computer Science & Engineering, Ammini College of Engineering, Kannampariyaram, Mankara, Palakkad, Kerala, India
In association with
International Journal of Scientific Research in Science and Technology

Gradual Class Evolution Detection Using Class Based Ensembles

J Linita Lyle*, Soumya Balan P, Prof. Leya Elizabeth Sunny

Department of Computer Science and Engineering, M A College of Engineering Kothamangalam, Kerala, India

ABSTRACT

The recent advances in hardware and software have enabled the capture of different measurements of data in a wide range of fields. These measurements are generated continuously and in a very high fluctuating data rates. Mining data streams is concerned with extracting knowledge structures represented in models and patterns in non- stopping streams of information. The research in data stream mining has gained a high attraction due to the importance of its applications and the increasing generation of streaming information. Mining data streams is concerned with extracting knowledge structures represented in models and patterns in non-stopping streams of information. Data Stream classification poses major challenges than classifying static data because of several unique properties of data streams such as infinite length, concept drift, concept evolution and feature evolution. While extensive work has been done in the area of concept drift, concept evolution, a phenomena that induces concept drift has gained little recognition. Class evolution basically focuses on 3 aspects: the phenomenon of class emergence, disappearance and reoccurrence and is an important research topic for data stream mining. Most of the previous works implicitly regard class evolution as a transient change, which is not true for many real-world problems as in many real world applications class evolution is a gradual process. A class-based ensemble approach, namely Class-Based ensemble for Class Evolution (CBCE), is adopted to handle class evolution. By maintaining a base learner for each class and dynamically updating the base learners with new data, CBCE can rapidly adjust to class evolution. A novel under-sampling method for the base learners is used to handle the dynamic class- imbalance problem caused by the gradual evolution of classes. Based on the above concepts of gradual class evolution, a dataset containing records of tweets made in twitter is evaluated at different time stamps by converting the unstructured, dynamic dataset into a more compact form, to evaluate and analyse concept evolution.

Keywords : Data Stream Mining, Concept Drift, Class Evolution

I. INTRODUCTION

The intelligent data analysis has passed through a number of stages. Each stage addresses novel research issues that have arisen. Statistical exploratory data analysis represents the first stage. The goal was to explore the available data in order to test a specific hypothesis. With the advances in computing power, machine learning field has arisen. The objective was to find computationally efficient solutions to data analysis problems. Along with the progress in machine learning research, new data analysis problems have been addressed. Due to the increase in database sizes, new algorithms have been proposed to deal with the scalability issue. More over machine learning and statistical analysis techniques have been adopted and modified in order to address the problem of very large databases. Data mining is that interdisciplinary field of study that can extract models and patterns from large amounts of information stored in data repositories.

With the rapid development of incremental learning and online learning, mining tasks in the context of data stream have been widely studied. Generally, data stream mining refers to the mining tasks that are conducted on a

(possibly infinite) sequence of rapidly arriving data records.

The analysis of huge volumes of data is recently the focus of intense research, because such methods could give a competitive advantage for a given company. For contemporary enterprises, the possibility of making appropriate business decisions on the basis of knowledge hidden in stored data is one of the critical success factors. Similar interests in exploring new types of data are present in many other areas of human activity.

In many of these applications, one should also take into consideration that data usually comes continuously in the form of data streams. Representative examples include network analysis, financial data prediction, traffic control, sensor measurement processing, ubiquitous computing, GPS and mobile device tracking, user's click log mining, sentiment analysis, and many others. Data streams pose new challenges for machine learning and data mining as the traditional methods have been designed for static datasets and are not capable of efficiently analysing fast growing amounts of data and taking into consideration characteristics such as:

- Limited computational resources as memory and time, as well as tight needs to make predictions in reasonable time.
- The phenomenon called concept drift, i.e., changes in distribution of data which occur in the stream over time. This could dramatically deteriorate performance of the used model.
- Data may come so quickly in some applications that labelling all items may be delayed or sometimes even impossible.

Out of several tasks studied in data streams, supervised classification has received significant attention. It is often applied to solve many real life problems such as discovering client preference changes, spam filtering, fraud detection, and medical diagnosis to enumerate only a few. The aforementioned speed, size and evolving nature of data streams pose the need for developing new algorithmic solutions. In particular, classifiers dedicated to data streams have to present adaptation abilities, because the distribution of the data in motion can change. To tackle these challenges, several new algorithms, specialized sliding windows, sampling methods, drift detectors and adaptive ensembles have been introduced in the last decade.

As the environment where the data are collected may change dynamically, the data distribution may also change accordingly. This phenomenon, referred to as concept drift, is one of the most important challenges in data stream mining. A data stream mining technique should be capable of constructing and dynamically updating a model in order to learn dynamic changes of data distributions, i.e., to track the concept drift.

For classification problems, concept drift is formally defined as the change of joint distribu tion of data, i.e., $p(x,y)$, where x is the feature vector and y is the class label. Over the past few decades, concept drift has been widely studied. The majority of the previous works focus on the concept drift caused by the change in class-conditional probability distribution, i.e., $p(x|y)$.

In comparison, class evolution, which is another factor that induces concept drift, has attracted relatively less attention. Briefly speaking, class evolution is concerned with certain types of change in the prior probability distribution of classes, i.e., $p(y)$, and usually corresponds to the emergence of a novel class and the disappearance of an outdated class. In some literature, class evolution is also called class-incremental learning or concept evolution.

We put forward a method by which data streams can be analysed more efficiently to detect class evolution.

The data sets is divided into independent units, pre-processed and structured in a way that makes mining tasks easier to be performed. By using this method, small and large datasets may be analysed to gain valuable information in fields like network analysis, medical diagnosis etc.

A. Class Based Ensemble For Class Evolution

Equation (6) suggests that the optimal classification strategy is to assign an example according to the likelihood that it belongs to a class. Therefore, a natural approach to this problem is to maintain a model for each class so that the likelihood can be explicitly estimated. For this reason, the CBCE approach is proposed. Each class-based model (CB model) is maintained for a

certain class ci and an example x is classified according to

$$\text{argmax CBMClassify}(x, CBM_i) \quad (1)$$

where the function CBMClassify returns the likelihood P(x| ci) or scores that can be used to estimate P(x| ci). Depending on the current class evolution state, the CBCE algorithm manages the CB models in mining tasks. Specifically, it may create a new CBmodel for a novel class, inactivate an outdated CB model for a disappeared class and re-activate the CB model when the class reoccurs again. Since the class conditional probability is also likely to change in a real-world data stream, the previously built model for a class could become invalid later. Hence, CBCE also involves a scheme to detect and handle the invalid CB model.

1) CLASS BASED MODEL

A class-based model is one that is specifically constructed for a certain class to get the likelihood (or related score) of a test example. A variety of models are possible candidates for a CB model, e.g., one-class classifier and clustering model. In this work, the CB model is implemented as a binary classifier that is able to output its classification posterior probability. In each CB model, with the one-versus-all strategy, the represented class is the positive class (+1) and the others are the negative one (-1) as a whole. According to Bayesian theory, the posterior probability P(+1|xt) for the positive class at time t is

$$P(+1|x_t) = \frac{P_t(+1)}{P_t(x_t)}. P_t(x_t| + 1) \quad (2)$$

where $Pt(xt)$ is the same for all classes. If the training data are balanced in CB models, P(+1) is a constant 1/2. In this condition, the posterior probability for positive class is proportional to the likelihood of the positive class, i.e., the specific class the CB model is maintained for. In other words, the probability can be used as the score to represent the likelihood for making decisions.

The positive and negative classes are likely to be imbalanced in a CB model. Although class-imbalanced problem has been intensively investigated, most previous studies focus on static class-imbalanced problems. In our case, the prior distribution may change over time, leading to a dynamic class-imbalanced problem. To address this issue, an under-sampling strategy is embedded in each CB model. The sampling probabilities for the positive and negative classes are different. As each CB model acts as an "expert" for its corresponding class, all of the examples received from this positive class are selected. The data size of the negative classes is usually larger than the positive one. Furthermore, the size of each class dynamically changes due to the gradual class evolution. These negative examples are sampled by under-sampling with a dynamic probability, which aims to select the negative data with the same size as the positive ones. Denoting wti as the prior probability of class ci at time t, the probability of sampling the negative examples for ci is calculated as

$$P_i = \min(w_t^i/(1 - w_t^i),\ 1) \quad (3)$$

In on-line learning, the underlying prior probability wti is hard to be observed. To quickly and accurately estimate wti, it is tracked by the time decay method as:

$$w_t^i = \beta w_{t-1}^i + (1 - \beta)1[y_t = c_t] \quad (4)$$

Where β (0 < β < 1) denotes decay factor, and 1[yt=ct] = 1 if yt, the true class label of xt is ci and 1[yt=ct] = 0 if yt, the true class label of xt is not ci.

To conveniently apply CBCE in practice, a constant decay factor is used for the prior probabilities of all classes. Since the estimated prior probability will be updated exponentially, it will quickly achieve its underlying value. The appropriate value for β is 0.9, which has been determined after comprehensive experiments.

In the CBCE framework, a CB model is required to provide its output in the form of score and can be updated on the-fly. Quite a few classical base leaners satisfy the first requirement, and logistic regression might be the model that has been mostly investigated with regard to the second issue. Hence, the online Kernelized Logistic Regression is employed in this work as the base learner. It should be noted that CBCE does not necessarily require to establish only one CB model for each class, and in some cases an ensemble model might be more suitable than a single model for a class. For example, if the minority class may comprise small disjuncts of data, a possibly better option for the CB

model is to employ cluster over-sampling techniques and build a model for each disjunct of data.

II. CLASS EVOLUTION ADAPTATION

Class evolution has three basic elements, i.e., the emergence of novel classes, the disappearance of outdated classes, and the reoccurrence of disappeared classes.

When a novel class ci emerges at time stamp t, CBCE first estimates its prior probability wti, and then initializes a new CB model CBMi for it. The prior probability is initially estimated after receiving the first two examples of this class. Denoting ExampleSize as the example size of the negative classes between these two examples, the prior probability is estimated as follows:

$$w_t^i = 1/(ExampleSize + 1) \qquad (5)$$

Based on the two examples of novel class and the negative examples between them, the CB model is initialized. Next, the CB model participates in classifying the subsequent data stream.

For class disappearance, the approach has to determine the disappearance when a class is shrinking; following this, its CB model should be managed to ensure not to affect the recognition of other classes. Since the evolution state is tracked in CB models, a sufficiently small prior probability threshold, e.g., $\beta 1000(\beta$ is the decay factor), can be used for disappearance confirmation. That is, if the class has been absent for 1000 consecutive time stamps, it is thus considered to have disappeared. The decision boundary of the CB model, as implemented by binary classifier, merely separates one class from another. In this case, if the class-conditional probability distribution changes or a novel class emerges on the boundary, the original CB model for the disappeared class would be inaccurate and also influence the novel class. Therefore, the CB model of the disappeared class is inactivated in classification. Besides, when a class is considered to have disappeared, its estimated prior probability is set to be 0, which also means its CB model is suspended for updating.

Class reoccurrence means that an example with the label of a disappeared class is received again. Effective handling of class reoccurrence could make use of past training efforts. Once class reoccurrence happens, the model re-estimates the prior probability in the same way as class emergence, and activates the CB model in classification.

B. DESIGN

The most creative and challenging phase of the system lifecycle is the system design. The term design describes a final system and the process by which it is developed. In system design, there is a movement from the logical to the physical aspects of the life cycle.

1) INPUT DESIGN

Input design is one of the most expensive phases of the operation of computerized system and is often the major problem of a system. The decisions made during the input design are: to provide cost effective method of input, to achieve the highest possible level of accuracy, to ensure that input is understood by the user.

2) Data Set Input

UDI TwitterCrawl Dataset, including 50 million tweets posted mainly from 2008 to 2011, is involved. Each record in this data set has its own time stamp and the order of examples in the data stream is completely genuine, without any modification. Since the hashtag roughly describes the tweet's topic, it was used as the class for each tweet record. If more than one hashtags exist in a tweet, one of them is selected randomly as its label.

Three tweet stream fragments from the whole tweet set are captured by selecting different topics as the classes of interest, i.e., tweet stream a, b, and c.

C. SYSTEM MODULES

The system developed has four modules.

- Initial Setup
- Update CB Model
- Class Evolution Adaptation
- Visualization

1) Initial Setup

Data pre-processing is an important step in the data mining process. Data-gathering methods are often loosely controlled, resulting in out-of-range values impossible data combinations missing values, etc. Analyzing data that has not been carefully screened for such problems can produce misleading results. Thus, the representation and quality of data is first and foremost before running an analysis. If there is much irrelevant and redundant information present or noisy and unreliable data, then knowledge discovery during the training phase is more difficult. Data preparation and filtering steps can take considerable amount of processing time. Data pre-processing includes cleaning, Instance selection, normalization, transformation, feature extraction and selection, etc. The product of data pre-processing is the final training set.

In this module the data stream is pre-processed and converted into a form suitable for performing the mining tasks. The stream is divided into smaller, more compact units.

2) Update CB Model

A class-based model is one that is specifically constructed for a certain class to get the likelihood (or related score) of a test example. A variety of models are possible candidates for a CB model, e.g., one-class classifier and clustering model. In this work, the CB model is implemented as a binary classifier that is able to output its classification posterior probability.

When a new example is received, every CB model will update the estimation of prior probability of its class. For the class that the currently received example belongs to, its CB model uses it for updating directly. For the other CB models, the example is first sampled with the dynamic sampling probability, and then used to update the models as a negative training example.

The learning procedure is summarized in Algorithm 1. When a new example is received, every CB model will update the estimation of prior probability of its class (lines 2 and 5). For the class that the currently received example belongs to, its CB model uses it for updating directly (line 3). For the other CB models, the example is first sampled with the dynamic sampling probability, and then used to update the models as a negative training example (lines 6 and 7).

3) Class Evolution Adaptation

Class evolution has three basic elements, i.e., the emergence of novel classes, the disappearance of outdated classes, and the reoccurrence of disappeared classes. When a novel class ci emerges at time stamp t, CBCE first estimates its prior probability wi, and then initializes a new CB model CBMi for it.

For class disappearance, the approach has to determine the disappearance when a class is shrinking; following this, its CB model should be managed to ensure not to affect the recognition of other classes.

Class reoccurrence means that an example with the label of a disappeared class is received again. Effective handling of class reoccurrence could make use of past training efforts.

For the inactivated CB model of a disappeared class, it can be used again for classification when an example with an old label arrives, which makes CBCE efficient.

This mechanism to deal with the three key components of class evolution is wrapped around each CB model, which equips CBCE to track gradually evolved classes effectively. The procedure of class evolution adaptation is summarized in Algorithm 2. Depending on the change of prior probability, class evolution behaviour can be determined. The active CB models are updated by the sampled data, and the inactive ones are stored additionally in case of class reoccurrence.

4) Visualization

A primary goal of data visualization is to communicate information clearly and efficiently via statistical graphics, plots and information graphics. Numerical data may are encoded using dots, lines, or bars, to visually communicate a quantitative message.

The distribution of data and the class behaviour of the data set are expressed using graphs and plots to effectively visualize, analyse and reason about data.

3) OUTPUT DESIGN

Output design generally refers to the results and information that are generated by the system. Outputs of a system can take various forms. The most common are reports, screens displays, printed form, graphical drawing etc. The output also vary in terms of their contents, frequency, timing and format. The users of the output, its purpose and sequence of details to be printed are all considered.

In this project the output is represented through the visualization module in the form of graphs.

III. RESULTS AND DISCUSSION

After getting the tweet streams, the text of each tweet is transferred into the TF-IDF vectors. 242, 247, 242 numerical features are generated, respectively, for the tweet streams a, b, c classes. It can be seen that class evolution may occur frequently in tweet stream. Besides, according to the visualization of tweet streams in the figures below, it can be observed that the class-conditional probability distribution also changes over time in tweet stream.

IV.CONCLUSION

The class-based framework adopted by the system has a number of advantages in comparison to the existing methods. First, since a CB model is specifically maintained for a certain class, it is flexible to be created or removed to adapt to class evolution. This also decouples the whole model, and makes each CB model simple and concentrate on a single class. Second, by using the CB model, only a few of base learners need to be maintained, equal to the number of classes. Third, for massive-volume data streams, the master-slave structure (CB model – ensemble strategy) and the chunk based processing of the learning system is also very convenient for parallelization and distributed implementation.

V. REFERENCES

[1] Y. Sun, K. Tang, L. L. Minku, S. Wang and X. Yao, "Online Ensemble Learning of Data Streams with Gradually Evolved Classes," in IEEE Transactions on Knowledge and Data Engineering, vol. 28, no. 6, pp. 1532-1545, June 1 2016.

[2] J. Gama, I. Zliobait_ e, A. Bifet, M. Pechenizkiy, and A.Bouchachia, "A survey on concept drift adaptation," ACM Comput. Surv.,vol. 46, no. 4, pp. 44:1–44:37, Mar. 2014.

[3] M. Masud, Q. Chen, L. Khan, C. Aggarwal, J. Gao, J. Han, and B. Thuraisingham, "Addressing concept-evolution in concept drifting data streams," in Proc. IEEE 10th Int. Conf. Data Mining, Dec. 2010, pp. 929–934

[4] M. M. Gaber, A. Zaslavsky, and S. Krishnaswamy, "Mining data streams: A review," SIGMOD Rec., vol. 34, no. 2, pp. 18–26, 2005.

[5] S. Wang, L. Minku, and X. Yao, "A learning framework for online class imbalance learning," in Proc. IEEE Symp. Comput. Intell.Ensemble Learn., Apr. 2013, pp. 36–45

[6] Heitor Murilo Gomes, Jean Paul Barddal, Fabricio Enembreck and Albert Bifet, "A Survey on Ensemble Learning for Data Stream Classification" in ACM Computing Surveys, Vol. 50, no. 2, April 2017

[7] A. Bifet, G. Holmes, B. Pfahringer, R. Kirkby, and R. Gavald. New ensemble methods for evolving data streams. In SIGKDD, pages 139–148, 2009.