ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 4, Issue 5, May 2015

A Proposed Approach to Classification and Novel Class Detection in Data Streams

Ms. Rimjhim Singh[1], Manoj. B. Chandak[2]

M.Tech Scholar, CSE Department, SRCOEM, Nagpur, India[1]

Professor and Head, CSE Department, Nagpur, India[2]

Abstract: In the field of data mining, data streams play an important role. In order to make intelligent use of data streams, first we need to handle them properly. The notation data stream itself represents the nature of such data. They possess the property of being dynamic and continuous in nature. That is the data keeps on changing the features and properties with time. Due to the above mentioned properties of data streams, various challenges are posed by them to researchers. These challenges are infinite-length, concept-evolution, feature-evolution and concept-drift. Infinite-length is due to the continuous nature of data. Concept-evolution is due to the new emerging classes. Concept-drift is due to the drifting concept of the stream and feature-evolution is there because of the changing features. These challenges have been well studied by various researchers and they have proposed various approaches to handle them. In this paper we also try to propose a strategy based on strings to handle the problems of infinite-length, concept-evolution and concept-drift.

Keywords: mining, novel, concept-drift, concept-evolution

I. INTRODUCTION

Advances in the field of data storage have enabled researches to store and handle huge amount of data in real time. This leads to data that may grow beyond limit and such data is usually referred to as stream data. While talking about mining data, we mean handling streams of data. Like classification is one of the problems that need to be solved in data mining. Because of the continuous and dynamic nature of data it becomes impossible to handle data streams with the strategies developed for handling static data. Due to the above mentioned properties, data streams pose different challenges to researchers. One of the major issues is infinite-length. Stream data may grow beyond limit and it is not feasible to store such huge amount of historical data. Hence, it becomes impractical to use this data for training our model. Stream data also experiences changes in the underlying concept. This occurs when features of various classes present in the data start drifting towards new features slowly and the classes undergo the conceptual change. Third challenge is posed when the stream starts facing new concepts. By new concepts we mean, new classes that are unknown to our system and stream start appearing in it at some later stage. For example, in case of intrusion detection, if we consider each type of attack as a class label. When altogether new type of attack will occur then it will be a new class label and we can say that concept-evolution is present. The instances belonging to these classes will not be classified by our system. Last one is feature-evolution, where new features start appearing in stream and older features start fading away. We need to handle all these challenges in order to make efficient use of data streams.

Various strategies have been proposed by researchers to handle these problems. Most of these strategies are not able to handle the issues efficiently. In order to handle infinite-length problem various incremental learning techniques have been proposed. These techniques divide the data into chunks and then train the model with the recent data chunks available. As concept-drift occurs due to the changes in the concept of stream with time, we need to detect these changes and update the model regularly with them. Different researchers have proposed techniques to update the model. Many of the proposed models assume that the number of classes present in the stream are fixed and continue with the classification procedure. But this is not the case with data streams. As we know that concept-evolution is due to the occurrence of new classes in the stream that are unknown to our system also. Hence such models will not be able to classify the instances belonging to new classes that will lead to inefficient classification. So the model needs to be trained on the recently occurring new classes in the stream. As an example, in case of intrusion detection we need to train our model on new class of attacks so that they are detected and classified at a later stage. Because of this, traditional classifiers don't work efficiently with data streams, as they are not able to mechanically detect the novel classes. Feature-evolution is also handled by some researchers. We can handle it by keeping track of the new features and their count occurring in the stream. We must also keep track of the old features that are fading away from the stream. Here we need to modify the feature vector of the model.

In this paper we develop a strategy to handle data streams. We first form an ensemble of model. Secondly, try to detect the outliers from the model. Third we try to separate out the outliers due to concept drift and handle them. Lastly, we try to get the outliers occurring due to concept-evolution and try to handle them.

II. RELATED WORK

Researchers have done a lot of work and given many strategies that can handle infinite-length and concept-drift effectively. But many of them fail to handle concept-

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 4, Issue 5, May 2015

evolution and feature-evolution efficiently. Various incremental approaches have been proposed in this direction. These approaches are basically of two types, namely single model incremental approach and hybrid-batch incremental approach. In single model incremental approach (proposed in [1]) use the concept of a single model. This model is continuously updated with the new features or data appearing in the stream data. The modification is done dynamically and regularly. But the main drawback of this approach is that the model becomes too vague and large. Its feature set expands up to a great extent and it loses its significance. Other approach, hybrid-batch incremental approach (proposed in [2], [3]), uses a collection of models. In this technique each model is generated by using batch learning process. Recent data chunk is used to generate a model and then the models are updated dynamically. This approach is more efficient as it is easy to update the individual models. The models that become obsolete at some stage are discarded and are replaced by the newer models. While handling data streams we must know that outliers occur due to three main reasons, namely, concept-evolution, concept-drift and noise. We not only need to detect these outliers but we also need to find the cause of their occurrence. If the cause is known, only then we can develop strategy to handle the outliers. Otherwise it might happen we classify an instance as an outlier occurring due to concept-evolution while it originally occurred due to concept-drift and vice versa. In this way misclassification might occur. And if this will happen then the false alarm rate of the classifier will be high.

Spinosa et al (explained in [4]) proposed an approach that is able to handle concept-drift, infinite-length and concept-evolution also. It detected the novel classes by making use of clustering techniques. They created a model by using the clustering and this model was encompassed by the hyper-sphere. If a cluster was obtained, if it lied outside the hyper-sphere and if it also contained considerable number of instances then it was said to be a novel class. This technique is a one class classifier that is it assumes only one class as normal class. Rest all other classes are considered as novel. Hence it had a drawback that it can't be used with data having multiple classes. It also assumed that the shape of normal class instances in the feature space is convex. But this is not the case in real time situations.

Also many of the strategies developed to detect novel classes are of two main types: parametric and non-parametric. Parametric approaches assume that data follows certain distribution and then by using the parameters of the normal data it calculates the parameters of the distribution. If an instance is found that doesn't follow the set distribution parameters, it is declared as an instance belonging to novel class. Hence such strategies are restricted to certain data distributions. Non-parametric techniques never make any assumptions about data distribution and hence are never restricted by data distribution. The technique that we have developed is also non-parametric. Also most of the techniques developed for novel class detection are unable to detect the presence of multiple novel classes simultaneously. But our technique can detect and handle multiple novel classes occurring simultaneously in the data

III. PROPOSED APPROACH

In the proposed approach we try to develop a technique that is based on strings to work with stream data. We try develop a classification model that is an ensemble of several models. Each sub-model will also work as a classification model.

While handling data streams we must know that in data streams we encounter two types of classes, namely, existing classes and novel classes. Let us assume that 'L' is an ensemble of classification models {M1, M2, M3, …. , Mn} that we have developed. Following are some definitions that are frequently used in our approach.

Definition 1: Existing Class: Assume there is a class named 'C'. 'C' is said to be an existing class if at least one of the models 'Mi' belonging to ensemble of models 'L' has been trained on class 'C'. Otherwise it is not an existing class.

Definition 2: Novel Class: Assume there is a class named 'N' in our data. If none of the models belonging to the ensemble of models 'L' is trained on 'L', then 'N' is said to be a novel class. In short a novel class is completely unknown to our system.

Definition 3: Outliers: Assume there is an instance named 'X' that is to be classified. If 'X' doesn't belong to any of the class belonging to model, say, 'Mi', then 'X' is an outlier with respect to model 'Mi' belonging to the ensemble.

Definition 4: F-Outlier: Assume 'X' is an instance that is to be classified. If 'X' is an outlier with respect to each and every model 'Mi' belonging to ensemble 'L', then 'X' is said to be 'F-Outlier'. That is 'X' is an outlier with respect to the complete ensemble of model 'L'

A. Training Phase

This is the first step towards building the classification technique. In our technique, we use a fixed size data chunk. We divide the data into fixed size chunks. Each chunk contains 2000 instances in it. Totally we form three chunks of equal size. After that we apply the k-Medoid clustering technique to these chunks individually. By applying clustering to one data chunk we generate one classification model 'Mi' of our ensemble. In this we generate an ensemble containing three classification models. We apply k-Medoid clustering technique in our system because it works with data that contains outliers. Models are built from the recent data chunk. We then store the summaries of the clusters obtained. In this we store the number of clusters belonging to each model 'Mi' and the set of words 'Si' defining each cluster or belonging to each cluster.

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 4, Issue 5, May 2015

Ensemble of model classifies the test instance as follows: Assume 'X' to a test instance that is to be classified. First of all it will be sent to each and every model 'Mi' of the ensemble 'L'. It will be compared to each cluster belonging to each model of the ensemble. If 'X' will be found belonging to an existing class 'C' then it will classified by the model containing the class 'C'. But if it is detected as an outlier (OUT) by all of the models contained in the ensemble then it will be classified as final outlier (F-OUT).

B. Outlier Detection

After generating the three classification models we take next step to detect the outliers. For getting the outliers of the ensemble we make use of the stored summaries of the clusters that were in the form of number of clusters and the set of words 'Si' of the clusters. Assume if 'X' is an instance that is to be classified or is to be detected as an outlier.

We first obtain the set of words S(X) of the instance 'X' and compare it with the set of words 'Si' of the class 'Ci' belonging to model 'Mj'. If the set S(X) matches with the set of words 'Si' of any of the class 'Ci' belonging to odel 'Mj', then it is classified accordingly. Otherwise it is detected as an outlier to the model 'Mj'. In this way we obtain the outliers (OUT_j) with respect to each and every model.

In the second step we try to obtain the final outliers of the ensemble of models. Here we generate an array of outliers (OUT_j) with respect to every model 'Mj'. We then see whether an instance present in one outlier vector is also contained in the other two outlier vectors also. If so then it is declared as final outlier with respect to the ensemble 'L'. we store all such final outliers in vector 'FOUT'.

Algorithm1. F_OUT

Input: Models Mi and instances 'X'.

Output: FOUT (Vector containing outliers of the model).

1: For each model 'Mi' in M
2: If $S(X) \in Cj^{'} Mi$
3: Append name of class 'C_j' to test instance 'X'.
4: else
5: Add 'X' to OUT_i vector.
6: End if.
7: End for.
8: FOUT = Intersection operation (OUT_1, OUT_2…., OUT_i)
9: End algorithm.

C. Concept-Drift

1) Detection of Concept-Drift **:**

Once we obtain the FOUT vector, we must know that outliers present in it are generated due to three main reasons. These three reasons are outliers concept-drift, concept-evolution and noise. Our next step is to separate the outliers based on the cause of their occurrence. First of all we try to obtain the outliers due to concept-drift and try to handle them. We try to detect it in the following manner. For an instance OUT_k belonging to FOUT, we first retrieve its set of words 'S_k ', then we compare 'S_k' with the set of words of various clusters belonging to model 'Mi' by performing the operation of set intersection on S_k and set of words 'S_j' of various classes 'C_j' . We then compare the obtained result against the set of words 'S_k' of instance. If more than 50% of words instance are present in the cluster word set 'S_i' then we deduce that the outlier is present due to concept-drift in the data. We store such outliers in the array named CONDRIFT.

2) Handling Concept-Drift:

After detecting concept-drift, our next task to handle it properly. Here we obtain the set of words 'S_k' of each instance in CONDRIFT. We then perform the operation of set difference on the word set 'S_k' and the set of words 'S_i' of the class 'C_i' from which the instance has drifted. The resultant obtained contains the new features that have arrived recently in the class or the drifting features. These new features are stored in a vector named DRIFTWORD along the information about the original class 'C_i'. Then we apply the unique operation on DRIFTWORD vector to obtain the unique drift words. In the next step we construct a matrix CHKMAT. Initially the values in the matrix are set to one. The rows of the matrix CHKMAT denote the unique drift words and the columns of the matrix denote the probable classes 'C_j' of the model 'Mj'. For each occurrence of the unique drift word 'W_m' belonging to class 'C_j', we increment the matrix value CHKMAT [m, j] by 1. We then set a threshold value. If the value CHKMAT [m, j] exceeds the threshold value then the unique drift word 'Wm' is appended to the feature set of the class 'C_j'. In this way the new drifting features are added to the original model.

D. Concept-Evolution

1) Detection of concept-evolution :

Here we again take into consideration the FOUT vector. If the instance OUT_k belonging to the FOUT vector does not qualify the above criteria set for concept-drift, i.e. If the instance does not belong to any of the class 'C_j' belonging to any of the model 'Mj' of the ensemble, then we deduce that the reason its occurrence is concept-evolution. We store these instances in the vector CONEVOLUTION. The main criteria that needs to be fulfilled is that more than 50% words of the instance in the FOUT must not match with the set of words of any of the class.

2) Handling Concept-Evolution:

When we try to handle concept-evolution we not only need to find the novel class. We also need to detect the occurrence of multiple novel classes. We make use of CONEVOLUTION vector. We know that this vector contains the instances belonging to one or more novel classes. We simply apply the k-Medoid clustering technique to the CONEVOLUTION vector. By doing so, we obtain the different novel classes present in the vector. At the end we simply add the summaries of the data to one of the classification models that were generated initially.

DOI 10.17148/IJARCCE.2015.45123

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 4, Issue 5, May 2015

In this way the ensemble is updated. We try to present the algorithms in the next full implementation paper.

IV. DATA SETS

Our algorithm requires that the data is not multi-valued and multi-labeled. The data instance must belong to one class only.

A. 4 University Data Set:

Initially we began our work with the above data set. It was present in the form of HTML pages. We applied preprocessing to it and obtained the data in the tabular form. After applying preprocessing we found that the data obtained was multi-valued and multi-labeled. This was against our assumption. Hence we could not continue with this data set.

B. NASA Aviation Safety Reporting System:

NASA ASRS dataset maintains a record of various accidents taking place in the air industry. This data set can be downloaded from the official website of NASA. The data was already present in the tabular form. With each accident there is an anomaly associated with it. Like aircraft problem: critical, aircraft problem: less severe etc. this event anomaly can be considered as a class. But our data was multi-valued and multi-labeled also. So first we made attempts to remove multi-valued and multi-labeled data to meet the requirement of our algorithm. We then found that there were instances that had incomplete information. We also removed such data from the data set to make it more proper and informative. Our dataset finally contained six existing classes and two novel classes. It contained concept-drift and concept-evolution also.

V. CONCLUSION AND FUTURE SCOPE

We in our approach are presenting a string based approach to classify the data streams. We are able to handle concept-drift, infinite-length and concept-evolution. We are not able to handle feature-evolution that efficiently. We are also able to detect multiple novel classes present in the data simultaneously. Other researchers have used the strategy based on distances. These are quite different from our strategy. We will be presenting our algorithms along with the results once they are tested and developed efficiently. in future we can also work to make the chunk size variable. We can also work to handle feature-evolution effectively.

REFERENCES

[1] C.C. Aggarwal, J. Han, J. Wang, and P.S. Yu, "A Framework for On-Demand Classification of Evolving Data Streams," IEEE *Trans. Knowledge and Data Eng.*, vol. 18, no. 5, pp. 577-589, May 2006.

[2] M.M. Masud, J. Gao, L. Khan, J. Han, and B.M. Thuraisingham, "Classification and Novel Class Detection in Data Streams with Active Mining,".

[3] Y. Yang, X. Wu, and X. Zhu, "Combining Proactive and Reactive Predictions for Data Streams," *Proc. ACM SIGKDD 11th Int'l Conf. Knowledge Discovery in Data Mining*, pp. 710-715, 2005.

[4] E. J. Spinosa, A. P. de Leon F. de Carvalho, and J. Gama. Cluster-based novel concept detection in data streams applied to intrusion detection in computer networks. *In ACM SAC, pages* 976–980, 2008.

[5] OLINDDA: A Cluster Based Approach for Detecting Novelty and Concept-Drift in Data Stream by Eduardo J. Spinosa, Andr´e Ponce de Leon F.de Carvalho, Jo ao Gama in *ACM Symposium of Applied Computing SAC'07*.

[6] B. Wenerstrom and C. Giraud-Carrier, "Temporal Data Mining in Dynamic Feature Spaces," *Proc. Sixth Int'l Conf. Data Mining (ICDM)*, pp. 1141-1145, 2006.

[7] M.M. Masud, J. Gao, L. Khan, J. Han, and B.M. Thuraisingham, "Classification and Novel Class Detection in Concept-Drifting Data Streams under Time Constraints," *IEEE Trans. Knowledge and Data Eng*,vol. 23, no. 6, pp. 859-874, June 2011.

[8] M.M. Masud, J. Gao, L. Khan, J. Han, and B.M. Thuraisingham, "Classification and Novel Class Detection in Feature Based Stream Data*," IEEE Trans. Knowledge and Data Eng* , vol. 25, no. 7, July 2013.

[9] M.M. Masud, J. Gao, L. Khan, J. Han, and B.M. Thuraisingham, "Integrating Novel Class Detection with Classification for Concept-Drifting Data Streams*," IEEE Trans. Knowledge and Data Eng* , vol. 25, no. 7, July 2009.

[10] M.M. Masud, Q. Chen, J. Gao, L. Khan, J. Han, and B.M. Thuraisingham, "Classification and Novel Class Detection of Data Streams in a Dynamic Feature Space," *Proc. European Conf. Machine Learning and Knowledge Discovery in Databases (ECMLPKDD)*, pp. 337-352, 2010.

[11] M.M. Masud, Q. Chen, L. Khan, C. Aggarwal, J. Gao, J. Han, and B.M. Thuraisingham, "Addressing Concept-Evolution in Concept-Drifting Data Streams," *Proc. IEEE Int'l Conf. Data Mining (ICDM)*, pp. 929-934, 2010.

[12] A Review of Method of Stream data classification through Optimized Feature Evolution Process by Archana Bopche, Malti Nagle and Hitesh Gupta International *Journal Of Engineering And Computer Science ISSN:2319-7242*Volume 3 Issue 1 January, 2014 Page No.3778-3783.