RESEARCH ARTICLE OPEN ACCESS

Mining Data Stream for Novel Class Detection Using Classification Techniques

P. Geetha Krishnaja
PG Scholar, Software Engineering
Department of Information Technology
Gokaraju Rangaraju Institute of Engineering and Technology
Telangana – India

ABSTRACT
Data stream classification poses many challenges to the data mining community. In this paper, we address four such major challenges, namely, infinite length, concept-drift, concept-evolution, and feature-evolution. Since a data stream is theoretically infinite in length, it is impractical to store and use all the historical data for training. Concept-drift is a common phenomenon in data streams, which occurs as a result of changes in the underlying concepts. Concept-evolution occurs as a result of new classes evolving in the stream. Feature-evolution is a frequently occurring process in many streams, such as text streams, in which new features (i.e., words or phrases) appear as the stream progresses. Most existing data stream classification techniques address only the first two challenges, and ignore the latter two. In this paper, we propose an ensemble classification framework, where each classifier is equipped with a novel class detector, to address concept-drift and concept-evolution. To address feature-evolution, we propose a feature set homogenization technique. We also enhance the novel class detection module by making it more adaptive to the evolving stream, and enabling it to detect more than one novel class at a time. Comparison with state-of-the-art data stream classification techniques establishes the effectiveness of the proposed approach.
Keywords:- Data Stream, Concept-Evolution, Novel Class, Outlier

I. INTRODUCTION

DATA stream classification has been a widely studied research problem in recent years. The dynamic and evolving nature of data streams requires efficient and effective techniques that are significantly different from static data classification techniques. In this paper it address four major challenges of data stream namely, infinite length, concept-drift, concept-evolution, and feature-evolution. Since a data stream is theoretically infinite in length, it is impractical to store and use all the historical data for training. Concept-drift is a common phenomenon in data streams, which occurs as a result of changes in the underlying concepts [3],[4],[5]. Concept-evolution occurs as a result of new classes evolving in the stream. Feature-evolution is a frequently occurring process in many streams, such as text streams, in which new features (i.e., words or phrases) appear as the stream progresses.

The novel class detection process consists of three steps. First, a decision boundary is built during training. Second, test points falling outside the

decision boundary are declared as outliers. Finally, the outliers are analyzed to see if there is enough cohesion among themselves (i.e., among the outliers) and separation from the existing class instances. But Masud et al[2] did not address the feature-evolution problem. The feature-evolution problem is addressed in which also addressed the concept-evolution problem. However, both have two drawbacks. First, the false alarm rate (i.e., detection of existing classes as novel) is high for some data sets. Second, if there is more than one novel class, they are unable to distinguish among them. In this work, we propose a superior technique for both outlier detection and novel class detection to reduce both false alarm rate and increase detection rate. Our framework also allows for methods to distinguish among two or more novel classes. Here claim four major contributions in novel class detection for data streams.

First, I am going to propose a flexible decision boundary for outlier detection by allowing a slack space outside the decision boundary. This space is controlled by a threshold, and the threshold is adapted cto detect novel class instances using the discrete Gini Coefficient. With this approach, it is able to distinguish different causes for the appearance of the outliers, namely, noise, concept-drift, or concept-evolution. Here derive an analytical threshold for the Gini Coefficient that identifies the case where a novel class appears in the stream. Here empirically show the effectiveness of this approach. Third, apply a graph-based approach to detect the appearance of more than one novel class simultaneously, and separate the instances of one novel class from the others. Finally, my proposed approach addresses the feature evolution problem on top of the enhancements discussed above. This is the work that proposes these advanced techniques for novel class detection and classification in data streams and addresses feature-evolution. Here apply my technique on a number of benchmark data streams including Twitter messages, and outperform the state-of-the-art classification and novel class detection techniques.

II. RELATED WORKS

Most of the existing data stream classification techniques are designed to handle the efficiency and concept-drift aspects of the classification process[6],[7]. Each of these techniques follows some sort of incremental learning approach to tackle the infinite-length and concept-drift problems. There are two variations of this incremental approach. The first approach is a single-model incremental approach, where a single model is dynamically maintained with new data. For example, incrementally updates a decision tree with incoming data, and the method in incrementally updates micro clusters in the model with the new data. The other approach is a hybrid batch-incremental approach, in which each model is built using a batch learning technique. However, older models are replaced by newer models when older models become obsolete some of these hybrid approaches use a single model to classify the unlabeled data whereas others use an ensemble of models.

The advantage of the hybrid approaches over the Single model incremental approach is that the hybrid approaches require much simpler operations to update a model (such as removing a model from the ensemble). My proposed approach not only addresses the infinite length and concept-drift problems but also concept-evolution and feature-evolution. Another category of data-stream classification technique deals with concept-evolution, in addition to addressing length and concept-drift. Spinoza et al apply a cluster-based technique to detect novel classes in data streams. Their approach builds a normal model of the data using clustering, defined by the hyper sphere encompassing all the clusters of normal data. This model is continuously updated with stream continuously to reduce the risk of false alarms and missed novel classes. Second, here apply a probabilistic approach progression. If any cluster is formed outside this hyper sphere, which satisfies a certain density constraint, then a novel class is declared. However, this approach assumes only one "normal" class, and considers all other classes as "novel." Therefore, it is not directly applicable to multiclass data stream classification, since it corresponds to a "one-class" classifier. Furthermore, this technique assumes that the topological shape of the normal class instances in the feature space is convex. This may not be true in real data. Katakis et al.[8] propose a feature selection technique for data streams having dynamic feature space.

EXISTING SYSTEM

The problem of concept-evolution is addressed in only a very limited way by the currently available data stream classification techniques. We investigate this problem in this paper, and propose improved solutions. Our current work also addresses the feature-evolution problem in data streams, such as text streams, where new features emerge and old features fade away. Most existing data stream classification techniques address only the first two challenges, and ignore the latter two.

III. PROPOSED WORK

To make this paper self-contained, should briefly describe the existing novel class detection technique proposed in [2].

The data stream is divided into equal sized chunks. The data points in the most recent data chunk are first classified using the ensemble. When the data points in a chunk become labeled (by human experts), that chunk is used for training. The basic steps in classification and novel class detection are as follows: Each incoming instance in the data stream is first examined by a outlier detection module to check whether it is an outlier. If it is not an outlier, then it is classified as an existing class using majority voting among the classifiers in the ensemble. If it is an outlier, it is temporarily stored in a buffer. When there are enough instances in the buffer, the novel class detection module is invoked. If a novel class is found, the instances of the novel class are tagged accordingly. Otherwise, the instances in the buffer are considered as an existing class and classified normally using the ensemble of models. The ensemble of models is invoked both in the outlier detection and novel class detection modules. The outlier detection process utilizes the decision boundary (to be explained shortly) of the ensemble of models to decide whether or not an instance is outlier. This decision boundary is built during training (see Section 3.1). The novel class detection process computes the cohesion among the outliers in the buffer and separation of the outliers from the existing classes to decide whether a novel class has arrived. The following sections discuss the training and classification phases more elaborately.

A.Training Phase

The training data are clustered using K-means and the related summary of each cluster is saved as "pseudo point". A k-NN-based classifier is trained with the training data. Rather than storing the r w training data, K clusters are built using a semi-supervised K-means clustering, and the cluster summaries (mentioned as pseudo points) of each cluster are saved. These pseudo points constitute the classification model. The summary contains the centroid, radius, and frequencies of data points belonging to each class.

Storing the Cluster Summary Information

For each cluster, we store the following summary information in a data structure called pseudo point:

1. Weight: Total number of data points in the cluster;
2. Centroid: The cluster centroid is the middle of a cluster.
3. Radius: Distance between the centroid and the farthest data point in the cluster; and
4. Mean distance: The mean distance from each point to the cluster centroid.

The radius of a pseudo point is equal to the distance between the centroid and the farthest data point in the cluster. The raw data points are discarded after creating the summary. Therefore, each model Mi is a collection of K pseudo points. A test instance xj is classified using Mi as follows: Let h 2 Mi be the pseudo point whose centroid is nearest from xj. The predicted class of xj is the class that has the highest frequency in h. The data point xj is classified using the ensemble M by taking a majority vote among all classifiers.

Each pseudo point corresponds to a "hypersphere" in the feature space with a corresponding centroid and radius. The decision boundary of a model Mi is the union of the feature spaces encompassed by all pseudo points h 2 Mi. The decision boundary of the ensemble M is the union of the decision boundaries of all models Mi 2 M. Once a new model is trained, it replaces one of the existing models in the ensemble. The candidate for replacement is chosen by evaluating each model on the latest training data, and selecting the model with the worst prediction error. This ensures that we have exactly L models in the ensemble at any given point of time. In this way, the infinite length problem is addressed because a constant amount of memory is required to store the ensemble. The concept-drift problem is addressed by keeping the ensemble up-to-date with the most recent concept.

B. Classification and Novel Class Detection Each instance in the most recent unlabeled chunk is first examined by the ensemble of models to see if it is outside the decision boundary of the ensemble. If it is inside the decision boundary, then it is classified normally (i.e., using majority voting) using the ensemble of models. Otherwise, it is declared as an F-outlier, or filtered outlier. The main assumption behind novel class detection is that any class of the data has the following property. If there is a novel class in the stream, instances belonging to the class will be far from the existing class instances and will be close to other novel class instances. Since F-outliers are outside the decision boundary, they are far from the existing class instances. So, the separation property for a novel class is satisfied by the F-outliers. Therefore, F outliers are potential novel class instances, and they are temporarily stored in a buffer buf to observe whether they also satisfy the cohesion property. The buffer is examined periodically to see whether there are enough F-outliers that are close to each other. This is done by computing the following metric, which we call the q-neighborhood silhouette coefficient, or q-NSC. To understand q-NSC, we first need to define the concept of q; c-neighborhood. Definition 1 (c; q-neighborhood).

IV. NOVEL CLASS DETECTION PROPOSED APPROACH

My proposed technique applies the Lossless feature space conversion for feature-evolving streams, and also enhances the existing novel class detection technique in three ways, which are 1) outlier detection using adaptive threshold, 2) novel class detection using Gini coefficient, and 3) simultaneous multiple novel class detection. Before describing the improvements, we briefly outline the overall novel class detection process. A. Overview Algorithm 1 sketches the proposed novel class detection approach. The input to the algorithm is the ensemble M and the buffer Buf holding the outliers instances. At first, we create K0 clusters using K-means with the instances in Buf (line 2), where K0 is proportional to K, the number of pseudo points per chunk (line 1).

Then each cluster is transformed into a pseudo point data structure, which stores the centroid, weight (number of data points in the cluster) and radius (distance between the centroid and the farthest data point in the cluster). Clustering is performed to speed up the computation of q-NSC value. If we compute q-NSC value for every F-outlier separately, it takes quadratic time in the number of the outliers. On the other hand, if compute the q-NSC value of the K0F-outlier pseudo points (or O-pseudo point), it takes constant time. The q-NSC value of a O-pseudo point h is the approximate average of the q-NSC value of each instance in h This is computed as follows: First, we define c; q(h) in terms of a O pseudo point h.

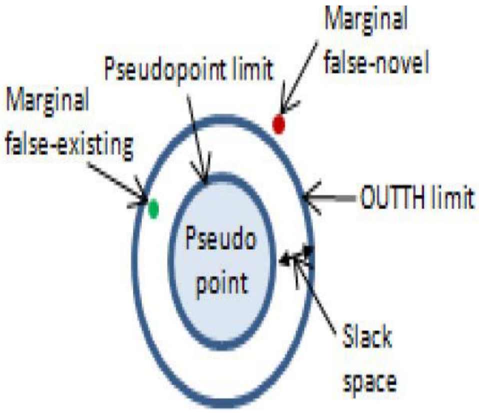
Algorithm 1. Detect-Novel(M,Buf) Input: M: Current ensemble of best L classifiers Buf: Buffer temporarily olding F-outlier instances

Output: The novel class instances identified, if found

1: Ko <- (K*|Buf|/S) //S = chunk size K = clusters per chunk
2: H<- K-means(Buf,K_0) //create K_0 O-pseudo points
3: for each classifier Mi Є M do
4: tp <- 0
5: for each cluster h Є H do
6: h.sc <- q-NSC(h)
7: if h.sc > 0 then
8: tp + = h.size //total instances in the cluster
9: for each instance x Є h.cluster do x:sc<-max(x.sc,h.sc)
10: end if
11: end for
12: if tp > q then vote++
13: end for
14: if vote == L then //found novel class, identify novel instances
15: Xnov <- all instance x with x.sc > 0
16: for all x Є Xnov do
17: x.ns Nscore(x)
18: if x.ns > Ginith then N list N list [x
19: end for 20: Detect-Multinovel(N list)
21: end if

OUTLIER DETECTION USING ADAPTIVE THRESHOLD

In this the algorithm takes the latest labeled instance x and the current OUTTH as input. It checks if x was a false-novel instance. This means that x belongs to an existing class but was falsely detected as a novel class instance. If x is false novel, then it must have been an F-outlier. Therefore, inst_weight(x) < OUTTH.



If the difference OUTTH -inst_weight(x) is less than a small constant∈, then we call x as a marginal false-novel instance. If x is found to be a marginal false-novel instance, then OUTTH is increased so that further instances like this do not fall beyond the decision boundary. Therefore, OUTTH is decreased by a small value (δ). This increases the slack space beyond the surface of a hypersphere. then x is a novel class instance but was wrongly identified as an existing class instance by a narrow margin, then the process to decrease the slack space (increase OUTTH) has to be made. This can be done by increasing OUTTH by a small value. The marginal constraint is applied to avoid drastic changes in OUTTH value. The value of OUTTH is not changed, if the test instance is NOT a marginal false-novel or false existing instance. On the other hand, if x is a marginal false-existing instance.

false existing instance.

Algorithm 2

Adjust-threshold(x, OUTTH)
Input: x: most recent labeled instance
OUTTH: current outlier threshold
Output: OUTTH: new outlier threshold
1: if false-novel(x) && OUTTH - inst_weight(x) < ∈ then
2: OUTTH -= δ //increase slack space
3: else if false-existing(x) && inst_weight(x) - OUTTH < ∈ then
4: OUTTH += δ //decrease slack space
5: end if

Novel Class Detection Using Gini Coefficient

After detecting the F-outlier instances using the OUTTH value discussed in the previous section, we compute the q-NSC(x) value for each F-outlier instance x using (1). If the q-NSC(x) value is negative, we remove x from consideration, i.e., x is regarded as an existing class instance. For the remaining F-outliers, q-NSC(.) is within the range [0, 1]. Now, we compute a compound measure for each such F-outlier, called Novelty score or Nscore.

$$\text{N Score}(x) = \frac{1-inst\ weight\ (x)}{1-minweight}\ \text{q-NSC}(x); \quad (2)$$

where minweight is the minimum inst_weight among all F-outliers having positive q-NSC. Nscore contains two parts: The first part measures how far the outlier is away from its nearest existing class pseudo point (higher value - greater distance). The second part measures the cohesion of the F-outlier with other F-outliers, and the separation of the F-outlier from the existing class instance. Note that the value of Nscore(x) is within [0, 1]. A higher value indicates greater likelihood of being a novel class instance. The distribution of Nscore(x) can be characterized by the actual class of F-outlier instances. In other words, by examining the distribution of Nscore(x), we can decide about the novelty of the F-outlier instance, as explained below. We discretize the Nscore(x) values into n equal intervals (or bins), and construct a cumulative distribution function (CDF) of Nscore. Let yi be the value of the CDF for the i^{th} interval. We compute the discrete Gini Coefficient G(s), for a random sample of yi, as follows

$$G(s) = \frac{1}{n}(n+1-2(\frac{\sum_{i=1}^{n}(n+i-i)y_i}{\sum_{i=1}^{n} y_i})); \quad (3)$$

The main intuition is that if the value of a variable takes a wide range of values, then (3) should be able to distinguish this case from the case where the variable is confined within a very short range of values. The former case occurs for Nscore(x) if there is a mix of data in the outliers (i.e., both concept-drift and concept-evolution), and the latter occurs if the outliers are either mostly existing class or mostly novel class. Let us consider three different cases and examine the behavior of G(s) in each case.

Case 1: All Nscore(x) are very low, and fall in the first interval. Therefore, $y_i = 1$ for all i. So G(s) becomes:

$$G(s) = \frac{1}{n}(n+1-2(\frac{\sum_{i=1}^{n}(n+i-i)}{n})) = 0$$

(after simplification). Note that this case occurs when all F-outliers actually belong to the existing classes.

Case 2: All Nscore(x) are very high, and fall in the last interval. Therefore, $y_i = 1$ and $y_i = 0$ for all i<n. So G(s) becomes: $G(s) = \frac{1}{n}(n+1-2(\frac{1}{1})) = \frac{n-1}{n}$. Note that this case occurs when all F-outliers actually belong to the novel class.

Case 3: Finally, we consider the case where Nscore(x) is evenly distributed across all the intervals. In this case $y_i = 1/n$ for all i. So G(s) becomes:

$$G(s) = \frac{1}{n}(n+1-2(\frac{\sum_{i=1}^{n}(n+i-i)i}{\sum_{i=1}^{n} i}))$$

$$= \frac{1}{n}(n+1-2(n+1-\frac{\sum_{i=1}^{n} i2}{\sum_{i=1}^{n} i}))$$

$$= \frac{1}{n}(-n-1-2(\frac{n(n+1)(2n+1)}{6}\ \frac{2}{n(n+1)}))$$

$$= \frac{n+1}{n} + \frac{2(2n+1)}{3n})) \quad = \frac{n-1}{3n}$$

Note that this case occurs if the distribution is mixed, i.e., noise, concept-drift and possibly some novel class instances. By examining the three cases, we can come up with a threshold for Gini Coefficient to identify a novel class as follows:

- If $G(s) > \frac{n-1}{3n}$, declare a novel class and tag the F-outliers as novel class instances.
- If G(s) = 0, classify the F-outliers as existing class instances.
- If $G(s) \in (0, \frac{n-1}{3n})$, filter out the F-outliers falling in the first interval, and consider rest of the F-outliers as novel class.

V. CONCLUSION

Here propose a classification and novel class detection technique for concept-drifting data streams that solve four major challenges, namely, infinite length, concept-drift, concept-evolution, and feature-evolution. The existing novel class detection techniques for data streams either do not address the feature-evolution problem or suffer from high false alarm rate and false detection rates in many scenarios. Here first discuss the feature space conversion technique to address feature-evolution problem. Then, we identify two key mechanisms of the novel class detection technique, namely, outlier detection, and identifying novel class instances, as the prime cause of high error rates for previous approaches.

To solve this problem, i propose an improved technique for outlier detection by defining a slack space outside the decision boundary of each classification model, and adaptively changing this slack space based on the characteristic of the evolving data. We also propose a better alternative approach for identifying novel class instances using discrete Gini Coefficient, and theoretically establish its usefulness. Finally, i propose a graph-based approach for distinguishing among multiple novel classes. Here apply my technique on several real data streams that experience concept-drift and concept-evolution and achieve much better performance than existing techniques. My approach uses fixed chunk size S for training. Here do not use any drift detection technique to make the chunk size dynamic. Therefore, if there is no concept-drift, our approach will still build a new model for each chunk (i.e., one for each S instances). Besides, if there is an abrupt drift, our approach will take some time to adjust to it. However, it could have used adopted some dynamic approach using drift detection technique but our present work emphasizes mainly on concept-evolution. Here it can consider the drift detection issue in the future to make our approach more dynamic and robust.

REFERENCES

[1] M.Masud "Classification and adaptive Novel class Detection of feature evolving data stream" vol.25,july 2013.

[2] M.M. Masud, J. Gao, L. Khan, J. Han, and B.M. Thuraisingham, "Integrating Novel Class Detection with Classification for Concept- Drifting Data Streams," Proc. European Conf. Machine Learning and Knowledge Discovery in Databases (ECML PKDD), pp. 79-94, 2009.

[3] H. Wang, W. Fan, P.S. Yu, and J. Han, "Mining Concept-Drifting Data Streams Using Ensemble Classifiers," Proc. ACM

SIGKDD Ninth Int'l Conf. Knowledge Discovery and Data Mining, pp. 226-235, 2003.

[4] J. Gao, W. Fan, and J. Han, "On Appropriate Assumptions to Mine Data Streams," Proc. IEEE Seventh Int'l Conf. Data Mining (ICDM), pp. 143-152, 2007.

[5] J. Kolter and M. Maloof, "Using Additive Expert Ensembles to Cope with Concept Drift," Proc. 22nd Int'l Conf. Machine Learning (ICML), pp. 449-456, 2005.

[6] G. Hulten, L. Spencer, and P. Domingos, "Mining Time-Changing Data Streams," Proc. ACM SIGKDD Seventh Int'l Conf. Knowledge Discovery and Data Mining, pp. 97-106, 2001.

[7] W. Fan, "Systematic Data Selection to Mine Concept-Drifting Data Streams," Proc. ACM SIGKDD 10th Int'l Conf. Knowledge Discovery and Data Mining, pp. 128-137, 2004.