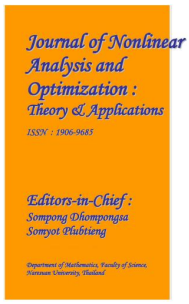
Journal of **N**onlinear **A**nalysis and **O**ptimization
Vol. 15, Issue. 1, No.15 : 2024
ISSN : **1906-9685**



Paper ID: ICRTEM24_111 **ICRTEM-2024 Conference Paper**

EXPLORING MACHINE LEARNING FOR STREAMING DATA: CURRENT STATUS, OBSTACLES, AND POTENTIAL

[#1]GADDAPAARA SWETHA, *Assistant Prfoessor,*
[#2]B.VEERA PRATHAP, Associate Professor,
Department of Computer Science & Engineering,
KLR COLLEGE OF ENGINEERING AND TECHNOLOGY, PALWANCHA,TS.

ABSTRACT: Terms like as "incremental learning," "online learning," and "data stream learning" are commonly used to characterize learning algorithms that update their models in response to a continuous stream of input without reprocessing it. Numerous papers have tackled this topic, either directly or indirectly, as the Velocity and Volume components of big data processing. Before existing methodologies can be effectively applied to real-world problems, a number of challenges must be overcome in light of the industry's current requirements. Our research seeks to clarify open problems in academia and industry, as well as the connections between the state-of-the-art in related fields. We pay special attention to topics that were not thoroughly covered in earlier position and survey papers. The goal of this study is to stimulate thought and clarify the available research routes by highlighting the linkages between diverse subfields and, where applicable, making recommendations for future directions.

KEYWORDS: Machine Learning, Real-time, Anomaly detection.

1. INTRODUCTION

When compared to the preceding decade, both the quantity of data sources and the rate of collection are increasing. These properties prompted the development of numerous machine learning approaches for data streams. The current shift in traditional machine learning methods is akin to the upcoming transfer of these approaches, which are on the verge of migrating from academic laboratories to the marketplace. In the present day, there is a movement that advocates for the invention and refinement of methodologies that go beyond the limits of learning algorithms. It emphasizes the development of strategies for managing data preparation and other practical activities, which are just as important as the development of learners who can adjust to new situations and successfully accomplish assignments.

Furthermore, in order to define the precise conditions and methods under which specific approaches developed for hypothetical situations can be used in real-world scenarios, it is necessary to conduct a thorough reevaluation of the underlying assumptions. Researchers in the field of machine learning have developed a variety of supervised learning works, particularly classification works, in the context of streaming data. The vast majority of these works addressed the subject of concept drifts, a phrase that ref
ers to temporal fluctuations in the distribution of the data under discussion. The majority of these studies focus on a single challenge: developing ways to maintain the accuracy of a decision model capable of learning and forgetting concepts in a

progressive manner. In recent years, supervised learning has been relegated to the sidelines in favor of other activities. This is primarily due to the need to handle more general issues that arise in the actual world.

Currently accessible projects include data stream clustering, pattern mining, multi-output learning, anomaly detection, feature selection, semi-supervised learning, and novel class recognition. These are only a few of the projects accessible. However, some localities did not make as much success as others. A substantial amount of study has been conducted on detecting and recovering drift in streaming data using labels that are instantly (and totally) available. Despite this, there has been little investigation into the efficiency of drift detection systems with streaming data. This is especially true when labels are never obtained in their full or arrive with a significant delay (as in semi-supervised and unsupervised learning). Previous scientific publications have addressed some of these difficulties, as well as the study choices they offer. Krempl et al. address essential problems such as evaluating data stream algorithms, transitioning from algorithms to comprehensive decision support systems, and protecting personal identifiable information. Big data surveys frequently encounter difficulties when it comes to machine learning for data streams.

These issues are connected to the Volume and Velocity components of the traditional 3Vs of big data: Volume, Variety, and Velocity. Nonetheless, there is substantial debate over the best methodology for learning from streaming data; hence, numerous abstractions and approaches will be used in accordance with the unique application and research team. During their discussion of continuous learning, Stoica et al. claim that reinforcement learning is a better technique for gathering knowledge from heterogeneous situations in which changes are expected to occur. This is an illustration of their point. Several evaluations and surveys focus on specific methodologies or activities, such as ensemble learning, rule mining for data streams, and activity recognition. Finally, here are some instances of previous investigations.

The goal of this research is to provide a current and detailed description of the current state and potential avenues for improvement in the field of machine learning that is specifically tailored to data streams. The investigation will concentrate on the data streams. Within the scope of our contributions, we explore topics that have not received appropriate attention in earlier publications of a similar sort. As a result, we direct readers to scientific papers that provide a more detailed review of the primary themes at hand, with an emphasis on more complicated issues. To maintain brevity, the points covered are given in the order listed below. The first three sections of the article cover a variety of key topics, such as learning, adaptation, and preprocessing. In this section, we will look at algorithms, streaming data and the artificial intelligence challenges that come with it, and the tools that are currently available for evaluating machine learning in the context of streaming data. Each of these three phases brings the task to an end. Section 8 presents a succinct review of the most relevant results presented throughout the study. These findings allude to the obstacles encountered and the prospects for further research.

2. DATA PREPROCESSING

Machine learning-based solutions must include data preprocessing. To aid the building of machine learning models, practical issues frequently involve changes to unprocessed data, preprocessing procedures, and an additional sample of "relevant" data. Streaming situations can complicate even simple preprocessing operations like feature normalization. Certain statistical information about the data, such as the minimum and maximum values that a specific feature can achieve, is unknown from the outset. There are several ways for scaling and discretizing features, as detailed in the next subsections; nevertheless, when we move on to more complicated preprocessing, we frequently reach uncharted or unexplored area. Data preparation is

mostly undertaken for two purposes prior to the training of machine learning models.

1. To help learning algorithms digest data more efficiently;
2. To improve the learning process by selectively retaining or extracting the most relevant information.

The initial rationale is more limiting in nature, as certain algorithms will be unable to analyze data that is not properly formatted, particularly in terms of incompatible data types. Alternative procedures may be ineffectual in the absence of data standardization. Examples of such algorithms include stochastic gradient descent and distance-based algorithms (such as nearest neighbors). The next phase is governed by feature engineering, as accurate machine learning solutions frequently rely on painstakingly planned feature reduction, selection, or modification of raw data. A batch learning workflow consists of three independent stages: model preprocessing, fitting, and testing. This sequential process produces an executable fitted model that can be applied to new data sets. These steps are also necessary in a streaming context. The primary distinction is that streaming data needs continuous application of the complete pipeline while in use. As a result, each phase is condensed into a single digital procedure that necessitates rigorous coordination of pipeline activities and, ideally, eliminates the necessity for offline execution of specific tasks. García et al. highlighted the difficulty of implementing preprocessing methods in big data contexts. The researchers looked at how several big data frameworks (such as Hadoop, Spark, and Flink) manage these difficulties, as well as the methodologies utilized for various preprocessing tasks (such as discretization, feature selection, and instance selection).

Ramírez-Gallego et al. discuss pre-processing techniques for data streams, including challenges and experimental methods. Significant questions were raised concerning the following: The development of novel discretization techniques that outperform quantile-based approaches in the presence of concept drifts; the incorporation of recurrent drifts into concept drift preprocessing techniques; and the creation of techniques to address non-traditional challenges such as multi-label classification. This section emphasizes on topics and methodologies (such as summarization illustrations) that have not been thoroughly covered in previous publications. Additional parts cover a wide range of preparation-related topics, including the management of imbalanced data.

Feature transformation

Managing limited memory while streaming data is difficult because these streams supply large amounts of raw data that are frequently worthless when examined alone but become critical when combined. These aggregated data structures can be used to train a machine learning model or to conduct additional analyses, such as "What is the average age of all customers who visited a particular website within the last hour?" A "sketch" of data is a summary made to eliminate the need for storing and maintaining huge volumes of information. Data streams are reduced into probabilistic data structures known as sketches. They accomplish this by allowing any two sketches from different streams to be merged in a space-efficient manner into the sketch of the combined stream. When employing sketches, several sacrifices must be made: they need a limited amount of memory and processing time to create, and incorrect sketches may result in wrong information being obtained from them. Many summary concepts have been proposed in recent decades.

There are numerous procedures accessible, ranging from basic membership strategies like counting and Bloom filters to more advanced methods like CM-Sketches and ADA-Sketches. Bloom filters, a type of probabilistic data structure, use hash functions to determine if an element belongs to a set. CM-Sketch is a generalization of Bloom filters, which are used to calculate the frequency of individual elements in a stream. The use of sketches to handle data streams is becoming more widespread. Ten years ago, some questioned the utility of drawing in stream processing.

Gaber et al., for example, recommend utilizing principal components analysis or other dimensionality reduction methodologies in favor of other methods since they believe that this is a more environmentally friendly way to handle streams. Creative drawings have been proposed based on the creation of a meta-sketch from multiple sketches. The Slim-Fat Sketch (SF-Sketch), which outperforms individual illustrations, exemplifies this. Data stream machine learning methods may also make use of sketching. The Graphical Model Sketch is a memory-saving sketch used in Bayesian networks and Naïve Bayes classifiers. It is not clear how designs can be incorporated with other machine learning techniques.

Feature Discretization

The discretization approach uses intervals to divide numerical qualities into categorical ones. Discrete variables are often easier to manage than numeric variables, hence discretization can shorten computation times. It can also lessen the chance of overfitting by simplifying the feature space, depending on the application and prediction model utilized. The Partition Incremental Discretization algorithm (PiD) marked a significant milestone in target feature discretization from data streams. PiD discretizes numerical properties into two layers. Given the input data, the first layer computes a huge number of intervals, while the second layer uses the first layer's statistics to produce equal-frequency partitions.

Webb proposed two distinct feature decretization schemes: IDAW (where W represents windowing) and Incremental Discretization Algorithm (IDA). IDA uses random sampling to perform quantile-based discretization on the entire data stream, whereas IDAW keeps track of the most recent values for each attribute and discretizes those. Because IDAW must be updated more frequently than IDA, it requires greater computing resources. The ChiMerge discretization algorithm is more noise-resistant than previous strategies because it saves feature values in a binary search tree.

Pfahringer et al. compared various discretization algorithms for Hoeffding trees. Empirical studies revealed that the Gaussian approximation is the most accurate strategy in terms of accuracy and tree development. Finally, the emphasis on feature discretization should be on providing effective implementations that interact with other components of the streaming process, such as drift detection, evaluation, and classification systems, in a similar way to feature scaling.

Invalid entries handling

Invalid entries can occur due to a variety of factors such as unfamiliar formats, missing data, noise, and other issues. It depends on the specific software, methods, and circumstances utilized to determine the features of an incorrect input. It is possible that categorical characteristics present in a range of machine learning tools, such as scikit-learn, are considered incorrect not because of flaws inherent in the tools themselves, but because the software was designed to disregard them.

Despite the technological issues connected with inappropriate submissions, the most evident and well-documented risk is a lack of data. In addition to discussing the severity of the issue, Krempl and his colleagues investigated the best technique to handle missing data in the output feature. This second instance is one we'd prefer talk about in the context of our semi-supervised learning conversation, so we'll just focus on the input data in this section. The process of correcting missing values in batch learning typically involves the use of imputation methods.

The study of imputation methods for streaming data has been limited. The main reason for this is because the methods typically rely on examining the complete dataset before adding numbers. These methods can be used for batch learning, but not for streaming data. It is possible that mean and median imputation will fail when attempting to estimate these statistics for data in constant rotation. Concerns with the mean estimate were as follows: Imputation can be implemented with the assistance of a learner, allowing you to avoid doing aggregation calculations. It is possible to employ windowed K nearest neighbors to forecast the value of a missing feature in a specific

instance based on the values of the k nearest neighbors.

Dimensional reduction Dimensionality reduction focuses on retaining relevant patterns in incoming data for the sake of learning. The following part will address feature selection methodologies for data streams, as well as the limitations of these procedures. Furthermore, we look at dimensionality reduction methods like Principal Component Analysis (PCA) and Random Projections, focusing on their possible applications and limits. Mitliagkas and colleagues suggested a memory-constrained principal component analysis (PCA) approximation that employs sampling and sketching methodologies to calculate within acceptable error limits.

According to Yu et al, a one-pass randomized principal component analysis technique was created; however, it has only been evaluated on one picture dataset. In their paper, Zhou and colleagues introduced an incremental feature learning method that uses denoising autoencoders to determine the optimal amount of model complexity for real-time data. There is a significant group of methodologies used for reducing dimensions, which is text-specific procedures. Vowpal Wabbit provides significant hashing algorithms for dimensionality reduction in these cases. When traditional Bag-of-Words and n-grams are insufficient, hashing techniques can be used to process text input in streaming contexts. It is typical practice in streaming domains to not assume that learners have a complete lexicon of the domain, as new concepts may emerge throughout the learning process. Each word (feature) in the initial text input is converted into a key using the hash function, which is then used in the hashing trick.

Following that, this key is used to enhance counters in a position with less dimensions, and the learner is given the counters. This approach does not support reverse lookups, which include identifying the most important terms for predictions, which is a severe limitation. Hoeffding Trees are used in an ensemble model proposed by Pham et al. This model includes multiple different random projection techniques, including Bernoulli, Achlioptas, and Gaussian projections. It has been demonstrated that the method is competitive when compared to bagging techniques, and it has been tested using real data. It is required to undertake additional research on all of the methods mentioned in this area, particularly to evaluate the effects of notion drift. Despite the fact that the bulk of these technologies are one-pass solutions, they have been used in batch processing systems to manage datasets. In practical streaming scenarios, drifts in the source data may cause discrepancies in the feature transformation outcomes of random projections. To establish the appropriate classifier changes, a thorough examination is required.

Characteristics and Selection Criteria

The goal of the feature selection procedure is to identify which features are relevant to the learning task at hand. When compared to dimensionality reduction procedures, feature selection maintains the data's interpretability by retaining its original form. In the field of batch machine learning, feature selection is widely recognized for its ability to reduce the amount of computation required, shorten the amount of time spent computing, and improve the generalization rates of classification systems by reducing the likelihood of underfitting. When it comes to feature selection algorithms that are tailored for batch settings, a thorough examination of the entire dataset is required to determine which features are the most significant based on a goodness-of-fit metric. In contrast, the availability of additional data over time precludes this requirement from being met in streaming systems. Barddal et al. demonstrated that decision rules and Hoeffding trees are the fundamental forms of classification and regression systems used in data streams to gradually identify the most important aspects.

There is a need for discussion on the problem of gradually identifying the most important qualities in data streams. New strategies for the feature selection process must be developed in order to detect and respond to changes in feature relevance,

also known as feature drift. The evaluation of feature selectors is an important constraint that must be considered while choosing features for streaming applications. When making feature selection recommendations, a variety of factors must be considered. Over time, feature selectors have been evaluated using a variety of objective criteria, one of which is "ease of use," as well as quantitative aspects such as scalability and accuracy.

It is critical to analyze how feature selection techniques interact with various learners, even if each learner is provided with the same set of features. This happens because each student creates their own unique predictive model. It is critical to ensure that the feature selection process is precise, and that the subset of characteristics picked matches to the attributes deemed to be of true value. The feature selectors are expected to be "stable," meaning that they will consistently select the same features even when trained on different samples of data.

When performing batch learning, stability measurements are used to determine how well a specific set of features aligns across multiple data samples from the same distribution. To ease the process of learning a model from data, stable techniques with a predetermined collection of attributes are used. In the context of streaming, the tradeoff between selection precision and feature stability is still being debated. If the relative relevance of characteristics fluctuates over time, as is the case with feature drift, the feature selection technique will have to choose between accuracy and stability.

3. EVALUATION PROCEDURES AND DATA SOURCES

As the subject rises in popularity among professionals and students, it is critical to evaluate the applicability of the benchmark datasets and assessment criteria now in use in the field to address real-world challenges. Choosing appropriate benchmark data is critical to avoid making algorithm quality judgments based on empirical trials conducted on data that may not accurately represent real-world conditions. Current evaluation systems address issues such as cross-validation, inconsistent data, and time dependence, among others.

A simple No Change learner, for example, could be a useful reference model if the incoming data stream has temporal relationships. The learner consistently assigns the next label to the previous one, displaying performance comparable to sophisticated learning algorithms that generate complex models from incoming data. Zliobaite et al. (2015) presented the κ-temporal statistic, which uses the No Change learner as an evaluation measure. The lack of appropriate methodologies for assessing delayed labeled scenarios makes it difficult to evaluate streaming algorithms. As stated in section 3.2, using a delayed setting results in a considerable temporal discrepancy between the input data x and the ground-truth label y.

Depending on the circumstances, the delay can last anywhere from a few minutes or hours to several years. To test the quality of these solutions, just record the learner's prediction φy for x and compare it to y when y is available for inspection. Some systems, such as peer-to-peer lending and airline delay prediction, may aggregate the learner with the same x before y becomes available. This could increase the model's performance if more labels are retrieved and included in future model updates. It is preferable for the learner to enhance their performance by generating predictions when x is introduced, but it is unclear how improvement can be measured until y is delivered. Efforts have been made to resolve delayed labeling, although evaluation frameworks have only been established recently. 5.1 Data Benchmarks Data stream algorithms are commonly evaluated by comparing them to a benchmark that includes both real-world datasets and synthetic generators. Using synthetic data allows you to demonstrate the method's usefulness in a controlled situation for specific challenges such as idea drifts, concept evolution, feature drifts, and more. Real-world datasets are typically utilized to assess the method's applicability in real-world scenarios rather than

fictional settings.

Nonetheless, their implementation does not guarantee the absence of algorithmic errors. It is difficult to predict when and when idea drift happens in a real-world dataset. For example, the artificial streams in the STAGGER and SEA databases are seen as too simplistic and possibly obsolete. Some researchers utilize synthetic datasets to replicate real-world data streams, which may not be identical to the original data streams. The Pokerhand4 dataset, which was previously used to assess the effectiveness of streaming classifiers, fits both characteristics (it is not a stream and it is synthetic).

This is likely due to the dataset's large size. Nonetheless, it cannot be considered authentic or a representation of an uninterrupted flow of information. It is still operational without a clear rationale. Some fundamental properties of benchmark datasets, which may be viewed as real-world data streams, are sometimes overlooked despite their heightened importance. The dataset used in Electricity represents the energy market of the Australian New South Wales Electricity Market. Although data is acquired progressively, an offline preprocessing phase is typically used to normalize the features.

This method can aid specific algorithms while also revealing future patterns. Issues with assessment frameworks go beyond supervised learning in a streaming environment. There are disagreements over how algorithms designed to detect idea drift should be evaluated. Integrating inventive concept drift detection with a classification algorithm and observing classification performance is a frequent strategy for evaluating the idea drift detection method's capabilities. One disadvantage of this evaluation is that it is indirect, making it difficult to observe essential factors such as the latency between drift and detection, both of which are fundamental properties of the drift detection technique. A recent publication provides thorough information on the subject. Why don't we test the efficacy of stream learning algorithms on actual data streams? One probable factor is the difficulty in preparing sensor data. Although there is a large amount of data available, it is critical to convert it into a usable format. This typically entails converting a multivariate time series into a data stream. Another possible explanation is that it can be difficult to duplicate and configure valid data stream sources.

Handling authentic streaming data

The technique for sending data to the system is an important feature of data streaming in practical applications. High-latency data sources will cause the entire system to fail, and the learning algorithm will be unable to resolve the issue. In general, novices have a harder time understanding the data source for streaming data than they do for batch learning. It is critical that the system efficiently communicates new information to the learner without delay, rather than simply being a separate file or database. Apache Spark and Flink are two of the leading stream processing frameworks.

The Apache Spark development team has released a new application programming interface (API) called the Structured Streaming API. This API replaces the prior Spark Streaming API. Structured Streaming, like its antecedent, is mostly built on micro-batches. Instead of giving new data rows to the user code immediately, they are organized into compact and cohesive batches. This enables data modification, as truly incremental solutions can be challenging for both users and framework developers to implement successfully. The primary distinction between Spark Streaming and the new Structured Streaming API is that the latter uses the concept of an unlimited database and assumes that the streaming data is structured, allowing for SQL data manipulation.

4. CONCLUSION

Several difficulties with machine learning and streaming data have been rectified. We examine the flaws in the research conducted to address these challenges, which frequently yielded only partial success. While every topic covered in this work is essential, some have been thoroughly researched or have a higher

influence. Moving these forward in the near future will accelerate progress in the field. The objectives consist of the following:

- Examining the relationships between adaptive stream mining algorithms and other advances in artificial intelligence, such as reinforcement learning and recurrent neural networks;
- Identifying and categorizing changes in data patterns without labeled data as soon as possible; Developing adaptive learning approaches that account for verification delays; and
- Combining preprocessing methods that adjust raw data on a regular basis.

It is critical to develop software that facilitates the application of data stream mining techniques in real-world scenarios.Many frameworks have recently been built, and the community updates and maintains them on an ongoing basis.A complete examination of all elements of machine learning using streaming data will take several research. As a result, we only briefly touched on several aspects that will require greater examination in the future, such as unstructured data sources like text, age, pattern mining, and regression.

REFERENCES

1. Z.S.Abdallah, M.M.Gaber, B.Srinivasan, and S.Kr- is hna swamy.Activity recognition with evolving data streams: Areview.ACM Computing Surveys(CSUR), 51(4):71, 2018.
2. A.Agarwal, O.Chapelle,M.Dud´ık,andJ.Lang- ford. A reliable effective terascale linear learning sys- tem.The Journal of Machine LearningResearch, 15(1):1111–1133, 2014.
3. C. C. Aggarwal, J. Han, J. Wang, and P. S. Yu. A frame work for clustering evolving data streams. In International Conference on Very Large Data Bases (VLDB), pages 81–92, 2003.
4. C. C. Aggarwal and P. S. Yu. On classification of high- cardinality data streams. In SIAM International Con- ference on Data Mining, pages 802–813, 2010.
5. T. Al-Khateeb, M. M. Masud, L. Khan, C. C. Aggar-wal, J. Han, and B. M. Thuraising ham. Stream classi- faction with recurring and novel class detection using class-based ensemble. In ICDM, pages 31–40, 2012.
6. M.Armbrust,T.Das,J.Torres,B.Yavuz,S.Zhu, R. Xin, A. Ghodsi, I. Stoica, and M. Zaharia. Structured streaming: A declarative api forreal-timeappli- cations in apache spark. In International Conference on Management of Data, pages 601–613, 2018.M.Baena-Garc´ıa,J.delCampo-A´vila,R.Fidalgo,
7. Bifet, R. Gavald`a, and R. Morales-Bueno. Earlydrift detection method. 2006.
8. D. Barber.Bayesian Reasoning and Machine Learning. Cambridge University Press, 2012.
9. J. P. Barddal, H. M. Gomes, and F. Enembreck. Analyzing the impact of feature drifts in streaming learn- ing. In International Conference on Neural Informa- tion Processing, pages 21–28. Springer, 2015.
10. J.P.Barddal,H.M.Gomes,F.Enembreck,and B. Pfahringer. A survey on feature drift adaptation: Definition, benchmark, challenges and future direc- tions. Journal of Systems and Software,127:278–294, 2017.
11. R.Bardenet,M.Brendel,B.K´egl,andM.Se- bag. Collaborative hyper parameter tuning. In Inter- national Conference on Machine Learning,pages 199– 207, 2013.
12. M.Barreno,B.Nelson,R.Sears,A.D.Joseph,and J. D. Tygar. Can machine learning be secure? In ACM Symposium on Information, computer and communications security, pages 16–25, 2006.