Improving Data Stream Classification using Incremental Yeo-Johnson Power Transformation

Eduardo Tieppo, Jean Paul Barddal, Júlio Cesar Nievola
Programa de Pós-Graduação em Informática (PPGIa)
Pontifícia Universidade Católica do Paraná (PUCPR)
Curitiba, Brazil
{eduardo.tieppo, jean.barddal, nievola}@ppgia.pucpr.br

***Abstract*—Data transformation plays an essential role as a preprocessing step in learning models. Several classification techniques have premises about the underlying data distribution, such as normal distribution assumed in Bayesians classifiers. However, applying data transformation in a streaming setting requires processing an infinite and continuous flow of data. In this paper, we propose the Incremental Yeo-Johnson Power Transformation, a variant of the well-known batch Yeo-Johnson transformation that is tailored for streaming settings, i.e., it supports streaming data via statistical sampling and hypothesis testing. Experimental results show that our proposal achieves the same data normality as its batch counterpart. In addition, it improves the prediction performance of a data stream classifier based on Bayesian statistical models. Overall, learning models obtained 3 percentage points improvement.**

***Index Terms*—Data stream classification, Incremental power transformation, Incremental Yeo-Johnson**

I. Introduction

In data stream classification, data are continuously made available as an unbounded sequence. Thus, learning models need to process one instance at a time and adapt themselves on-the-fly to process the latest data and support unbounded streams [1].

Data stream classification problems are tackled with learning models based on a learning paradigm, such as symbolic, instance-based, or statistical, with distinct ideas of data representation and predictions strategies [2]. These prediction strategies are often based on data distribution assumptions. For instance, Bayesian models assume that data follow a normal distribution, and inferences are made with probability calculations using the Bayes Theorem [2].

Several studies have used data transformations to make input data more fitted to a given distribution and, consequently, to a learning model that uses this same distribution as a premise. There are examples from Engineering [3] to Medicine [4], and even general research in the machine learning field about how data transformation techniques affect the learning models' performance [5]. However, applying data transformation in streaming settings is not straightforward. As data streams are potentially unbounded, there is no full dataset to be transformed and supplied to the learning model.

Supported by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001.

In this study, we propose an incremental adaptation of the well-known Yeo-Johnson Power Transformation [6] for streaming settings. Traditionally, the Yeo-Johnson Power Transformation is applied considering a batch setup as it is applied to a fully known dataset, which is transformed into a more gaussian-like distributed dataset. More specifically, our proposal is to apply the Yeo-Johnson Power Transformation incrementally, one instance at a time, without prior knowledge about data, and adaptively along the data stream. Our proposal can be attached to a data stream learning model that uses a Gaussian (normal) data distribution as a premise, such as the well-known Naive Bayes method, to improve the prediction performance of this classifier when applied in a data stream scenario [2]. Therefore, the contributions of this study are as follows:

- We propose the incremental Yeo-Johnson Power Transformation, an incremental adaptation of the well-known Yeo-Johnson Power Transformation that does not require the entire dataset in advance, it is performed on an instance-basis, and it is adaptive along the stream.
- We apply the incremental Yeo-Johnson Power Transformation to two case studies on flat and hierarchical data stream classification and demonstrate the improvements from its application on predictive performance.

Furthermore, as a by-product, we publicize the source code and datasets used in experimentation for reproducibility.

The remainder of this paper is organized as follows. Section II describes studies related to data transformation in classification tasks and outlines related works. Section III describes the proposed incremental Yeo-Johnson Power Transformation. Section IV comprises the analysis performed to test our proposal, including its application in several streaming classification scenarios. Finally, Section V concludes this study and states envisioned future works.

II. Related Work

Data transformation is a common step in traditional machine learning models to improve the input adherence to an assumption, like data-normality [5]. Different usages of data transformation can be observed in theoretical aspects in [5], [7] and applied scenarios in [4], [8]. However, studies that use methods and algorithms for data transformation have been developed considering stationary data or assuming that

the whole dataset could be accessed at least once. Data stream classification requires an additional concern related to computational resources usage [1]. Nevertheless, current data transformation techniques that limit resources' usage focus only on data representation and prediction efficiency.

Research has been conducted on similar scenarios related to data stream mining and big data. For example, the authors in [9] applied the Box-Cox [10] transformation with automatic estimation of the power parameter but did not consider data stream classification scenarios. Also, authors in [5] proposed a Box-Cox information array to compute model parameters based on limited data but still process the whole dataset once. Therefore, these studies did not consider on-the-fly constraints imposed by streaming settings where data is potentially unbounded and, thus, instances need to be analyzed and discarded right after its processing [11].

Regarding learning systems, Bayesian statistical models stand out among the well-known techniques that can benefit from data transformations as their inductive process is based on Bayes' Theorem to calculate the probability of an instance belonging to a given class given the *a priori* probability of occurrence of the class considering the instance features. That is, Bayesian classifiers assume that data is normally distributed [2]. In addition, a Bayesian classifier, such as the well-known Naive Bayes [12], handles data stream classification problems since the learning model only needs to update summaries of data along with the data stream and compute the probabilities for each incoming instance [13], [14].

In data stream classification, incremental adaptations of Bayesian classifiers have been widely studied and applied in state-of-the-art algorithms [14], [15]; thus, it is of interest to provide a way to address data transformations also incrementally. For instance, an Incremental Gaussian Naive Bayes was proposed in [14] observing the data streaming constraints, and it demonstrated the advantages of using the Yeo-Johnson data transformation in the predictive performance of the classifier. However, the data transformation was performed considering a hypothetical complete view of the data streams, and thus, we tackle this limitation by applying the proposed incremental Yeo-Johnson Power Transformation on an instance-basis, and without prior data knowledge.

III. Incremental Yeo-Johnson Power Transformation

This section proposes the Incremental Yeo-Johnson Power Transformation fitted to work with data stream learning models. First, we describe the traditional Yeo-Johnson technique [6] and later discuss the adaptations needed to make the technique incremental. The traditional Yeo-Johnson Power transformation $\psi(\lambda, x)$ is defined as follows, where x stands for the input data and λ is the power parameter.

$$\psi(\lambda, x) = \begin{cases} \left\{(x+1)^{\lambda} - 1\right\}/\lambda & (x \geqslant 0, \lambda \neq 0) \\ \log(x+1) & (x \geqslant 0, \lambda = 0) \\ -\left\{(-x+1)^{2-\lambda} - 1\right\}/(2-\lambda) & (x < 0, \lambda \neq 2) \\ -\log(-x+1) & (x < 0, \lambda = 2) \end{cases} \quad (1)$$

Algorithm 1: Traditional application of Yeo-Johnson Power transformation.

input : D: a dataset with instances $\vec{x}$
L: a set of candidate λ
output: $\hat{D}$ – a dataset with transformed instances $\widehat{\vec{x}}$

1 $\hat{\lambda} \leftarrow \arg\max_{\lambda \in L} \ell(\lambda)$;
2 **foreach** $(\vec{x} \in D)$ **do**
3 $\quad \widehat{\vec{x}} \leftarrow \psi(\lambda, \vec{x})$; // Equation 1
4 $\quad \hat{D} \leftarrow \hat{D} \cup \widehat{\vec{x}}$;

The Yeo-Johnson Power transformation is based on Box-Cox transformation [10], [16] and was proposed to handle negative values. If x is positive, the Yeo-Johnson transformation converges to Box-Cox with $(x+1)$. If negative, it is the Box-Cox of $(-x+1)$ with power $(2-\lambda)$. Algorithm 1 shows the pseudocode for the application of Yeo-Johnson Power transformation [6]. Note that the estimated/optimal λ ($\hat{\lambda}$) can be obtained by maximizing the log-likelihood function ℓ of the transformation power parameter λ via Maximum Likelihood Estimation (MLE) using all data D. Also, if $\hat{\lambda}$ is known (or chosen) *a priori*, the arguments of the maxima (line 1) for ℓ can be omitted and the transformation can be performed per x using Equation 1.

To extend the traditional Yeo-Johnson Power transformation to handle unbounded data streams, the main concern regards the estimation of the optimal $\hat{\lambda}$ on the fly and the use of a strategy to ensure that this estimation will remain accurate over time. The proposed Incremental Yeo-Johnson Power transformation tackles these issues by applying sample size determination and hypothesis testing to (i) find the optimal $\hat{\lambda}$ based on a sample set, and (ii) check if two sample sets obtained from the data stream at different timestamps significantly differ, thus requiring a new $\hat{\lambda}$ estimation.

Figure 1 illustrates the incremental Yeo-Johnson Power Transformation in a streaming scenario. Let DS be a data stream supplying instances over time, N_i a user-given size for a chunk of DS with $i \in \{1, \ldots, \infty\}$, and SS a sample size estimated based on N_i. In practice, N_i represents the number of instances observed prior to sample size determination (SS_{N_i}) and new incoming instances of $N_i + 1$ trigger new statistical validations resulting in SS_{N_i+1}, and so forth. The

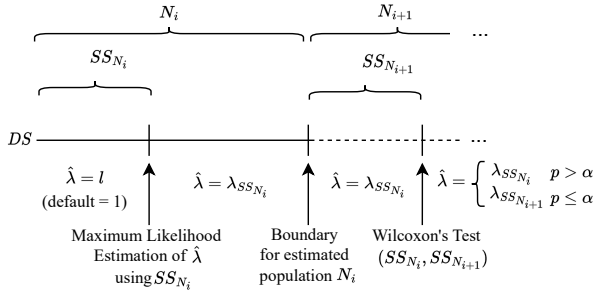

Fig. 1. Overview of the incremental Yeo-Johnson Power Transformation setup in a data stream scenario.

process initially assumes $\hat{\lambda} = l$, where $l = 1$ is a user-defined parameter (note that $\lambda = 1$ results in equivalent input and transformed data). The method buffers and uses SS_{N_i} instances to perform a Maximum Likelihood Estimation of $\hat{\lambda}$. The sample size determination follows Equation 2, where N_i stands for the population estimated size, z is the standard normal distribution, p and q are the complementary proportions for population, and d the error component of interval estimate.

$$SS_{N_i} = \frac{N_i z^2 pq}{d^2(N_i - 1) + z^2 pq} \tag{2}$$

The sample size determination considers population proportions and also a finite population correction; despite being parameterizable, p and q are always applied in this study with equal (0.5) proportions [17].

The Maximum Likelihood Estimation of $\hat{\lambda}$ is performed via Brent's algorithm [18] by minimizing the negative log-likelihood function of the transformation power parameter λ for the input data buffered (SS_{N_i}), resulting in an optimal $\lambda_{SS_{N_i}}$, which is used for the ongoing data transformation. Once the data stream DS has provided N_i instances, we reach the statistical warranty boundary for the estimated population N_i. In other words, the sample set buffered in SS_{N_i} may not be representative anymore for DS. Thus, we restart the process by buffering the newest SS_{N_i+1} instances to compare both sample sets SS_{N_i} and SS_{N_i+1}.

When SS_{N_i+1} is reached, a Wilcoxon's hypothesis test [19] verifies significant differences between SS_{N_i} and SS_{N_i+1} sample sets. The Wilcoxon's test is applied with a user-defined significance level α (default $= 0.05$), according to the protocol provided in [20]. Thus, the new value of $\hat{\lambda}$ is assigned as follows according to the $p\text{-}value$ returned from Wilcoxon's test. If both sample sets are significantly different, the optimal $\hat{\lambda}$ is recomputed for the newest input data buffered ($SS_{N_{i+1}}$); otherwise, the transformation continues with the $\lambda_{SS_{N_i}}$ computed on the previously MLE.

Algorithm 2 shows the pseudocode for the proposed Incremental Yeo-Johnson Power Transformation fitted for data stream classification. Note that, oppositely to traditional application of the Yeo-Johnson Power Transformation (cf. Algorithm 1), the algorithm receives a data stream (DS) providing instances $\vec{x}_t$ over time instead of a complete dataset. In addition, the algorithm receives the parameters needed to perform the sample size determination (see Equation 2) and also an initial *lambda* set $\vec{l}$ (default = 1).

The loop started on line 3, and the transformation performed on line 24 are equivalent to the ones on the traditional algorithm. The difference between both algorithms lies in how the $\hat{\lambda}$ estimate is performed. In the traditional algorithm, the estimation considers the entire dataset (cf. Algorithm 1, line 1), whereas, in the incremental strategy, this process is made along the data stream (lines 4-23).

First, the algorithm determines the sample size (line 1) and assigns the initial *lambda* to $\hat{\lambda}$. Note that $\hat{\lambda}$ is d-dimensional as the $\vec{x}_t$, with possibly distinct values for each d. Inside the loop, the algorithm buffers $|SS_{N_i}|$ instances (line 5) until the computed sample size and uses them to perform the first $\hat{\lambda}$ estimation (line 8). Note that, if provided, MLE considers a set L of candidate λ values. However, this is not critical when using Brent's algorithm, and the final estimation may not be a member of L [18]. When a new $\hat{\lambda}$ estimation is performed, it is necessary to clear or update any current learning model (line 9) since any new instance $\vec{x}_t$ is transformed into $\widehat{\vec{x}}_t$ using a different power parameter λ. The algorithm controls when the theoretical population size is reached (line 12) and starts to populate a new sample set with $|SS_{N_{i+1}}|$ instances (line 15). When $SS_{N_{i+1}}$ is full, the algorithm tests both initial SS_{N_i} and current $SS_{N_{i+1}}$ sample sets with the Wilcoxon's Test (line 18). If both sets are significantly different, the $\hat{\lambda}$ is re-estimated using $SS_{N_{i+1}}$ (line 20), and the current sample set is defined as the reference sample set.

Observe that a learning model can be appended by replacing line 25, and works with each single $\widehat{\vec{x}}_t$ over time without requiring changes in the learning method as the proposed method works as a preprocessing step.

Finally, we highlight two key aspects related to the application of the described Incremental Yeo-Johnson Power Transformation in the data stream classification scenario. First, every $\hat{\lambda}$ estimation requires a new learning model that, at least, is more concerned with the newest data since subsets

Algorithm 2: Incremental Yeo-Johnson Power Transformation for Data Stream Classification

input : DS: a data stream providing instances $\vec{x}_t$ over time $t \in \mathbb{N}$
L: a set of candidate λ
N, d, p, $q = p - 1$: sampling parameters
α: significance level
$\vec{l}$: initial *lambda* set
output: $\widehat{DS}$: an incrementally transformed data stream

1 $SS \leftarrow (Nz^2pq)/(d^2(N-1) + z^2pq)$;
2 $\hat{\lambda} \leftarrow \vec{l}$;
3 **foreach** $(\vec{x}_t \in DS)$ **do**
4 **if** $(|SS_{N_i}| < SS)$ **then**
5 $SS_{N_i} \leftarrow SS_{N_i} \cup \vec{x}_t$;
6 **else**
7 **if** $\neg(estimated)$ **then**
8 $\hat{\lambda} \leftarrow \arg\max_{\lambda \in L} \ell(SS_{N_i})$;
9 $Model \leftarrow \emptyset$;
10 $estimated \leftarrow \top$;
11 **else**
12 **if** $(t \bmod N = 0)$ **then**
13 $tested \leftarrow \bot$;
14 **if** $(|SS_{N_{i+1}}| < SS)$ **then**
15 $SS_{N_{i+1}} \leftarrow SS_{N_{i+1}} \cup \vec{x}_t$;
16 **else**
17 **if** $\neg(tested)$ **then**
18 $p\text{-}value \leftarrow Wilcoxon(SS_{N_i}, SS_{N_{i+1}})$;
19 **if** $(p\text{-}value \leq \alpha)$ **then**
20 $\hat{\lambda} \leftarrow \arg\max_{\lambda \in L} \ell(SS_{N_{i+1}})$;
21 $Model \leftarrow \emptyset$;
22 $SS_{N_i} \leftarrow SS_{N_{i+1}}$;
23 $tested \leftarrow \top$;
24 $\widehat{\vec{x}}_t \leftarrow \psi(\hat{\lambda}, \vec{x}_t)$; // Equation 1
25 $\widehat{DS} \leftarrow \widehat{DS} \cup \widehat{\vec{x}}_t$;

of transformed data with different λ present distinct distributions and may mislead the classifier. Also, note that this aspect justifies the use of Wilcoxon's test instead of promptly recomputing the optimal *lambda* with the newest buffer to try to avoid a change in $\hat{\lambda}$ and keep more representative data.

Second, depending on the error component of interval estimate (d), few instances can theoretically represent unbounded data. For example, assuming a confidence level of 95% and a margin of error of 2%, the sample size is bounded on 2,401 samples regardless of N size. Naturally, this is not the case in a data stream classification problem. Thus, we understand that λ recomputing can act as a response to concept drifts [21] on the underlying distribution of the data stream. In other words, the DS size (N) can represent a user-defined boundary to renew the statistical warranty about the sample set with a trade-off between responsiveness to concept drifts and maintaining historical data.

IV. Analysis

In this section, we describe the experiments performed to analyze our proposal. It comprises the experimental setup and two case studies concerning the application of our proposal in both flat and hierarchical data stream scenarios, including an analysis focused on the data stream transformations and their impact on the performance of the classifiers.

A. Experimental setup

We divided our experiments into data normality and prediction performance analysis, detailed below. Data normality was measured using the Shapiro–Wilk test of normality [22]. The test was performed on all data streams considering the original data, the transformed data with access to the complete data stream (known DS), and the data incrementally transformed with the proposed transformation. The prediction performance was measured on two distinct scenarios: a traditional data stream classification scenario, including well-known data stream sets, and a hierarchical data stream classification scenario, following the same protocol provided in [14] as a testbed.

In the traditional case study (hereafter referred to as "flat" data stream scenario), we used the well-known incremental Naive Bayes technique for data streams [2], [23] and measured the prediction performance of the classifier using the F-score [24]. In the hierarchical case study, we used the classifier proposed in [14], an incremental Gaussian Naive Bayes (GNB-hDS) fitted to work with hierarchical data streams and measured the prediction performance using the hierarchical F-Score (hF) [25].

In both case studies, the classifiers were applied to scenarios with the original data stream, the transformed data with access to the complete data stream (known DS), and the incrementally transformed data. All the experiments were performed using $N = 1.0 \times 10^7$, $d = (0.95, 0.02)$, $p = 0.5$, $\alpha = 0.05$ and $\vec{l} = 1$. Note that the population estimated size N was set up with the first power of ten that reaches the upper bound (2,401) of sample size determination (cf. Equation 2) and simulates a population estimated size always bigger than the known DS. Furthermore, the pair of values provided in d represent, respectively, the confidence level and the margin of error.

We report both F-score and hF metrics using the prequential test-then-train validation method, where each instance is used to test the model before it is used for training and updating [1]. The results obtained with the original data, the transformed data with access to the complete data stream (known DS), and the incrementally transformed data were compared using the Friedman test [26] to perform multiple comparisons in non-parametric data assuming a null hypothesis that there is no significant difference between the results. After the Friedman test and in case of the null hypothesis is rejected, we applied the Nemenyi *post-hoc* test [27] to identify significant differences between two specific sets. All significance tests considered a 95% confidence level according to the protocol provided in [20].

The experiments in this paper were performed using the Massive Online Analysis (MOA) Framework (2021.07) [23] (flat scenario) and Python 3.7 (hierarchical scenario). The scripts containing the implementations of all experiments are available at http://www.ppgia.pucpr.br/~jean.barddal/datasets/GNB-hDS-incYJ.zip.

B. Case study: Data Stream Classification

Table I describes the eight data stream sets used in our experiment concerning data stream classification, listing their number of instances, features, and classes. These data streams contain different features and domains, thus allowing the assessment of how our proposal works in different scenarios. Also, note that the Airlines, Asset Negotiation, Forest Covertype, and RTG data streams have categorical features. However, naturally, these features are not included in the transformations and do not interfere with the results obtained in all experiments.

Table II depicts the Shapiro-Wilk W Statistic for original, transformed, and incrementally transformed data streams. The W statistic is bounded by 1, and closer values to this upper bound represent data more fitted to a normal distribution. The average W Statistic obtained with the original data is 0.8132. In contrast, the averages obtained with both transformations (known DS and incremental) surpass the former by 4.44%. Note that the transformations less affected

TABLE I
FLAT DATA STREAMS USED IN THE EXPERIMENT.

Data stream	Instances	Features	Classes
Agrawal [28]	500,000	9	2
Airlines [29]	539,383	7	2
Asset Negotiation [30]	500,000	5	2
Forest Covertype [31]	581,012	54	7
Credit card [32]	284,807	30	2
Electricity [33]	45,312	8	2
Kaggle's GMSC [34]	150,000	10	2
RTG [35]	500,000	10	2

TABLE II
SHAPIRO-WILK W STATISTIC FOR ORIGINAL, TRANSFORMED, AND INCREMENTALLY TRANSFORMED FLAT DATA STREAMS.

Data stream	Original	Transformed (known DS)	Incrementally transformed
Agrawal	0.8186	0.8967	0.8963
Airlines	0.9169	0.9691	0.9441
Asset Negotiation	0.9321	0.9321	0.9321
Forest Covertype	0.9298	0.9803	0.9795
Credit card	0.8069	0.8564	0.7935
Electricity	0.7688	0.9345	0.7391
Kaggle's GMSC	0.3776	0.6211	0.6210
RTG	0.9551	0.9553	0.9553
Average	0.8132	0.8932	0.8576

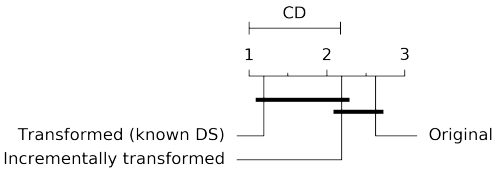

Fig. 2. Critical difference chart for Shapiro-Wilk W on flat streams.

TABLE III
F-SCORE (%) OBTAINED WITH ORIGINAL, TRANSFORMED, AND INCREMENTALLY TRANSFORMED FLAT DATA STREAMS.

Data stream	Original	Transformed (known DS)	Incrementally transformed
Agrawal	73.86	74.54	**74.64**
Airlines	61.38	**61.46**	60.79
Asset Negotiation	83.40	83.40	**83.44**
Forest Covertype	49.12	51.09	**51.20**
Credit card	55.54	55.58	**57.51**
Electricity	70.39	73.36	**73.63**
Kaggle's GMSC	51.09	**67.86**	67.82
RTG	61.25	61.24	**61.28**
Average	63.25	66.07	**66.29**

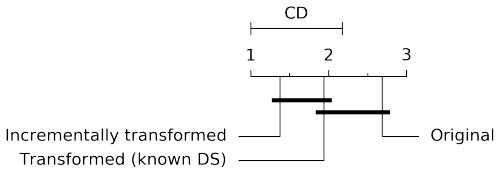

Fig. 3. Critical difference chart for F-score rates.

the data streams containing categorical features. For instance, Asset Negotiation has only one continuous feature against four categorical ones. The same can be observed on Airlines, Forest Covertype, or RTG data streams, where at least half of the features are categorical.

We also highlight the lowest rate between both transformations (known DS and incremental) in the Electricity data stream. This data stream comprises 45,312 instances, of which the first 17,424 instances (38.45%) originally did not have values for three continuous attributes, being filled up later with the average values from the other instances. Thus, the data sampling can not obtain representative information since the beginning of the stream provides only repeated values. Nevertheless, the W statistic is similar across all data stream sets, with Yeo-Johnson being applied with the data known *a priori* and incrementally with our proposal.

Figure 2 shows the critical difference (CD) chart for the Shapiro-Wilk W Statistic obtained with original data and both transformations on the flat data streams. Both transformations obtained significantly higher W statistics than those obtained with the original data streams, yet, no statistical difference is observed.

Regarding prediction performance, as aforementioned, we applied the incremental Naive Bayes classifier with the original data stream, the transformed data with access to the complete data stream (known DS), and the incrementally transformed data. Table III depicts the F-score (%) obtained in all experiments, where the highest values per data stream are highlighted in bold.

Overall, one can observe that both transformations improve the prediction performance of the Naive Bayes classifier. The average F-score obtained using the original data is 63.25%, and it is improved by 2.81% and 3.03% with transformed (known DS) and incrementally transformed data streams, respectively. We also highlight some noticeable improvements, as in the Electricity data stream, with an increase of about 3%, and in Kaggle's GMSC data stream, with more than 16% on both transformations. Also, the values obtained with both transformations are similar across all data streams, with an average difference of 0.40% favoring the incremental one. We submitted the F-score rates obtained with all data transformations to a Friedman test to check if the Naive Bayes classifier could achieve the same prediction results using traditional and incremental transformations.

Figure 3 shows the critical difference chart obtained after Friedman and Nemenyi tests for the F-score rates obtained. The Friedman test showed a statistical difference between the F-score rates and the *post-hoc* Nemenyi test only identified a difference between the performance of the classifier with and without data transformations but not between both transformations.

The performance of the classifier was significantly improved using both sets of transformed data streams. More importantly, there is no statistical difference between the performances using the traditional (known DS) and the incremental transformations.

C. Case study: Hierarchical Data Stream Classification

As previously described, we used the classifier (GNB-hDS) and the data streams proposed in [14] as a testbed for our proposal in a hierarchical data stream classification scenario [11]. We follow the same protocol provided in [14] in the classification experiment, as our proposal can be attached to any data stream classifier without modifications in the main process since it represents an additional on-the-fly data processing step.

Table IV depicts the 14 hierarchical data streams used in our testbed, listing their number of instances, features,

TABLE IV
HIERARCHICAL DATA STREAMS USED IN THE EXPERIMENT.

Data stream	Instances	Features	Classes
Entomology [36]	21,722	33	14
Ichthyology [36]	22,444	15	15
Insects-a-b [37]	52,848	33	6
Insects-a-i [37]	355,275	33	6
Insects-i-a-r-b [37]	79,986	33	6
Insects-i-a-r-i [37]	452,044	33	6
Insects-i-b [37]	57,018	33	6
Insects-i-g-b [37]	24,15	33	6
Insects-i-g-i [37]	143,323	33	6
Insects-i-i [37]	452,044	33	6
Insects-i-r-b [37]	79,986	33	6
Insects-i-r-i [37]	452,044	33	6
Insects-o-o-c [37]	905,145	33	24
Instruments [36]	9,419	30	31

TABLE V
SHAPIRO-WILK W STATISTIC FOR ORIGINAL, TRANSFORMED, AND INCREMENTALLY TRANSFORMED HIERARCHICAL DATA STREAMS.

Data stream	Original	Transformed (known DS)	Incrementally transformed
Entomology	0.7489	0.9517	0.9513
Ichthyology	0.9028	0.9839	0.9773
Insects-a-b	0.7236	0.9240	0.9204
Insects-a-i	0.7248	0.9272	0.9229
Insects-i-a-r-b	0.7268	0.9273	0.9250
Insects-i-a-r-i	0.7234	0.9269	0.9311
Insects-i-b	0.7239	0.9239	0.9254
Insects-i-g-b	0.7280	0.9273	0.9148
Insects-i-g-i	0.7227	0.9288	0.9166
Insects-i-i	0.7234	0.9269	0.9305
Insects-i-r-b	0.7250	0.9252	0.9249
Insects-i-r-i	0.7234	0.9269	0.9307
Insects-o-o-c	0.7416	0.9468	0.9462
Instruments	0.9689	0.9868	0.9865
Average	0.7577	0.9381	0.9360

and classes. Table V depicts the Shapiro-Wilk W Statistic for original, transformed, and incrementally transformed data streams. The average W Statistic obtained with the original data is 0.7577. In contrast, the averages obtained with both transformations (known DS and incremental) surpass 0.93.

As in the flat scenario, the W statistic is similar across all hierarchical data stream sets, with Yeo-Johnson being applied with the data known *a priori* and incrementally. The average W Statistic of traditional and incremental transformations differ by 0.0021.

Figure 4 shows the critical difference chart for the Shapiro-Wilk W Statistic obtained with original data and both transformations on the hierarchical data streams.

The tests showed a significant difference between Shapiro-Wilk W Statistic obtained with both transformations and the original data. Also, the tests confirmed that there is no difference between both transformations, thus, the Incremental Yeo-Johnson could achieve the same improvements in the data normality without accessing the complete data stream.

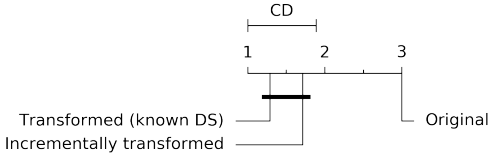

Fig. 4. Critical difference chart for Shapiro-Wilk W on hierarchical streams.

TABLE VI
$hF(\%)$ OBTAINED WITH ORIGINAL, TRANSFORMED, AND INCREMENTALLY TRANSFORMED HIERARCHICAL DATA STREAMS.

Data stream	Original	Transformed (known DS)	Incrementally transformed
Entomology	48.64	**53.87**	52.82
Ichthyology	46.82	**50.27**	49.72
Insects-a-b	81.11	81.90	**81.96**
Insects-a-i	80.88	**84.05**	84.03
Insects-i-a-r-b	81.42	83.48	**84.07**
Insects-i-a-r-i	81.57	83.40	**83.70**
Insects-i-b	80.55	82.28	**82.40**
Insects-i-g-b	81.53	81.42	**81.50**
Insects-i-g-i	80.40	83.16	**83.38**
Insects-i-i	80.90	83.05	**83.18**
Insects-i-r-b	78.57	79.58	**80.21**
Insects-i-r-i	81.61	83.45	**83.72**
Insects-o-o-c	64.14	**69.46**	69.38
Instruments	48.31	**49.93**	49.48
Average	72.60	74.95	**74.97**

Regarding prediction performance, Table VI depicts the Hierarchical F-score (hF) obtained with original, transformed, and incrementally transformed hierarchical data streams (highest values per data stream are highlighted in bold). As expected, both transformations improve the prediction performance of the Gaussian Naive Bayes classifier. The average hF obtained using the original data is 72.60%, and it is improved by more than 2% with transformed (known DS) and incrementally transformed data stream sets.

The values obtained with both transformations are similar across all data streams, with an average difference of 0.02% favoring the incremental one. We highlight that the classifier was even able to obtain better results using the incremental transformation than the traditional (known DS) transformation in 9 out of the 14 data streams, probably due to its ability to obtain a better λ estimation.

As described in the experimental setup subsection (IV-A), the hF rates obtained with all data transformations were statistically compared to check if the Gaussian Naive Bayes classifier could achieve the same prediction results using both transformations.

Figure 5 shows the critical difference chart obtained after Friedman and Nemenyi tests for the hF rates obtained.

Similar to the results obtained with flat data streams,

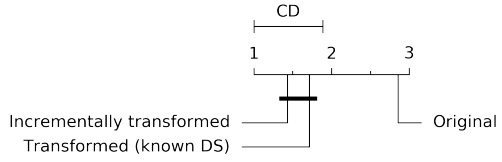

Fig. 5. Critical difference chart for hierarchical F-score rates.

Friedman and Nemenyi tests identified a difference between the performance of the classifier with and without data transformations but not between the traditional and the incremental transformations.

These results corroborate our claims that that our proposal can be applied to a data stream classification learning model as an attached data processing step without the need for a full view of the input data and can still improve the prediction performance of the classifier by reducing data skewness.

V. Conclusion

In this paper we introduced the Incremental Yeo-Johnson Power Transformation, an adaptation of the Yeo-Johnson Power Transformation. The proposal is suited for streaming scenarios as it performs the transformation on an instance-basis and adaptively along the stream.

Also, the incremental transformation can be appended to a learning model that uses data normality premises to improve data stream classification. Results showed that the incremental transformation obtains the same data normality as the traditional transformation and that Naive Bayes classifiers benefit in 3.03% in flat prediction rates and 2.36% in the hierarchical setting.

As future works, we plan to perform a deeper analysis of how different chunk sizes affect data normality and prediction performance on data streams with particular kinds of concept drifts. We are also interested in analyzing the impact of our proposal on decision trees (and their ensembles) that use Naive Bayes classifiers at their leaves.

References

[1] J. Gama, *Knowledge discovery from data streams*. Chapman and Hall/CRC, 2010.

[2] J. Han, J. Pei, and M. Kamber, *Data mining: concepts and techniques*. Elsevier, 2011.

[3] Y. Jiang, B. Cukic, and T. Menzies, "Can data transformation help in the detection of fault-prone modules?" in *Proc. of the Workshop on Defects in large software systems*, 2008, pp. 16–20.

[4] Y. Liang, A. Hussain, D. Abbott, C. Menon, R. Ward, and M. Elgendi, "Impact of data transformation: An ecg heartbeat classification approach," *Frontiers in digital health*, vol. 2, 2020.

[5] T. Zhang and B. Yang, "Box–cox transformation in big data," *Technometrics*, vol. 59, no. 2, pp. 189–201, 2017.

[6] I.-K. Yeo and R. A. Johnson, "A new family of power transformations to improve normality or symmetry," *Biometrika*, vol. 87, no. 4, pp. 954–959, 2000.

[7] A. Rahman, "Statistics-based data preprocessing methods and machine learning algorithms for big data analysis," *Int. J. of Artificial Intelligence*, vol. 17, no. 2, pp. 1–22, 2019.

[8] M. Castelli, F. M. Clemente, A. Popovič, S. Silva, and L. Vanneschi, "A machine learning approach to predict air quality in california," *Complexity*, vol. 2020, 2020.

[9] R. Maciejewski, A. Pattath, S. Ko, R. Hafen, W. S. Cleveland, and D. S. Ebert, "Automated box-cox transformations for improved visual encoding," *IEEE transactions on visualization and computer graphics*, vol. 19, no. 1, pp. 130–140, 2012.

[10] G. E. Box and D. R. Cox, "An analysis of transformations," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 26, no. 2, pp. 211–243, 1964.

[11] E. Tieppo, R. R. d. Santos, J. P. Barddal, and J. C. Nievola, "Hierarchical classification of data streams: a systematic literature review," *Artificial Intelligence Review*, pp. 1–40, 2021.

[12] C. M. Bishop, *Pattern recognition and Machine Learning*. Springer, 2006.

[13] F. Klawonn and P. Angelov, "Evolving extended naive bayes classifiers," in *Sixth IEEE International Conf. on Data Mining-Workshops*. IEEE, 2006, pp. 643–647.

[14] E. Tieppo, J. P. Barddal, and J. C. Nievola, "Classifying potentially unbounded hierarchical data streams with incremental gaussian naive bayes," in *Brazilian Conf. on Intelligent Systems*. Springer, 2021, pp. 421–436.

[15] A. Bifet, G. Holmes, B. Pfahringer, R. Kirkby, and R. Gavalda, "New ensemble methods for evolving data streams," in *Proc. of the 15th ACM SIGKDD international conf. on Knowledge discovery and data mining*, 2009, pp. 139–148.

[16] R. M. Sakia, "The box-cox transformation technique: a review," *Journal of the Royal Statistical Society: Series D (The Statistician)*, vol. 41, no. 2, pp. 169–178, 1992.

[17] W. W. Daniel and C. L. Cross, *Biostatistics: a foundation for analysis in the health sciences*. Wiley, 2018.

[18] R. P. Brent, *Algorithms for minimization without derivatives*. Courier Corporation, 2013.

[19] F. Wilcoxon, "Individual comparisons by ranking methods," in *Breakthroughs in statistics*. Springer, 1992, pp. 196–202.

[20] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *Journal of Machine Learning Research*, vol. 7, no. Jan, pp. 1–30, 2006.

[21] A. Tsymbal, "The problem of concept drift: definitions and related work," *Computer Science Department, Trinity College Dublin*, vol. 106, no. 2, p. 58, 2004.

[22] S. S. Shapiro and M. B. Wilk, "An analysis of variance test for normality," *Biometrika*, vol. 52, no. 3/4, pp. 591–611, 1965.

[23] A. Bifet, G. Holmes, B. Pfahringer, P. Kranen, H. Kremer, T. Jansen, and T. Seidl, "Moa: Massive online analysis, a framework for stream classification and clustering," in *Proc. of the first workshop on applications of pattern analysis*. PMLR, 2010, pp. 44–50.

[24] M. Sokolova, N. Japkowicz, and S. Szpakowicz, "Beyond accuracy, f-score and roc: a family of discriminant measures for performance evaluation," in *Australasian joint conference on artificial intelligence*. Springer, 2006, pp. 1015–1021.

[25] S. Kiritchenko and F. Famili, "Functional annotation of genes using hierarchical text categorization," *Proceedings of BioLink SIG, ISMB*, 01 2005.

[26] M. Friedman, "The use of ranks to avoid the assumption of normality implicit in the analysis of variance," *Journal of the American Statistical Association*, vol. 32, no. 200, pp. 675–701, 1937.

[27] P. Nemenyi, "Distribution-free multiple comparisons," in *Biometrics*, vol. 18. International Biometric Society, 1962, p. 263.

[28] R. Agrawal, T. Imielinski, and A. Swami, "Database mining: A performance perspective," *IEEE transactions on knowledge and data engineering*, vol. 5, no. 6, pp. 914–925, 1993.

[29] E. Ikonomovska, J. Gama, and S. Džeroski, "Learning model trees from evolving data streams," *Data mining and knowledge discovery*, vol. 23, no. 1, pp. 128–168, 2011.

[30] F. Enembreck, B. C. Ávila, E. E. Scalabrin, and J.-P. Barthès, "Learning drifting negotiations," *Applied Artificial Intelligence*, vol. 21, no. 9, pp. 861–881, 2007.

[31] J. A. Blackard and D. J. Dean, "Comparative accuracies of artificial neural networks and discriminant analysis in predicting forest cover types from cartographic variables," *Computers and electronics in agriculture*, vol. 24, no. 3, pp. 131–151, 1999.

[32] A. Dal Pozzolo, O. Caelen, R. A. Johnson, and G. Bontempi, "Calibrating probability with undersampling for unbalanced classification," in *IEEE Symposium Series on Computational Intelligence*. IEEE, 2015, pp. 159–166.

[33] M. Harries, *Splice-2 comparative evaluation: electricity pricing*. University of New South Wales, 1999.

[34] Kaggle - give me some credit. [Online]. Available: https://www.kaggle.com/c/GiveMeSomeCredit/

[35] P. Domingos and G. Hulten, "Mining high-speed data streams," in *Kdd*, vol. 2, 2000, p. 4.

[36] A. R. S. Parmezan, V. M. Souza, and G. E. Batista, "Towards hierarchical classification of data streams," in *Iberoamerican Congress on Pattern Recognition*. Springer, 2018, pp. 314–322.

[37] V. M. A. Souza, D. M. Reis, A. G. Maletzke, and G. E. A. P. A. Batista, "Challenges in benchmarking stream learning algorithms with real-world data," *Data Mining and Knowledge Discovery*, pp. 1–54, 2020.