2020 IEEE REGION 10 CONFERENCE (TENCON)
Osaka, Japan, November 16-19, 2020

Concept Drift Adaptation for Acoustic Scene Classifier Based on Gaussian Mixture Model

Ibnu Daqiqil Id
Graduate School of Interdisciplinary Science and Engineering in Health Systems
Okayama University
Okayama, Japan
daqiqil@s.okayama-u.ac.jp

Masanobu Abe
Graduate School of Interdisciplinary Science and Engineering in Health Systems
Okayama University
Okayama, Japan
abe-m@okayama-u.ac.jp

Sunao Hara
Graduate School of Interdisciplinary Science and Engineering in Health Systems
Okayama University
Okayama, Japan
hara@okayama-u.ac.jp

Abstract— **In non-stationary environments, data might change over time, leading to variations in the underlying data distributions. This phenomenon is called concept drift and it negatively impacts the performance of scene detection models due to them being trained and evaluated on data with different distributions. This paper presents a new algorithm for detecting and adapting to concept drifts based on combining the existing and new components Gaussian mixture model then merging it. The algorithm is equipped with a drift detector based on kernel density estimation enabling the algorithm to adapt to new data and generalize over old and new concepts well.**

Keywords— Concept Drift, Acoustic Scene Classification, Incremental Learning, Gaussian Mixture Model

I. INTRODUCTION

Enabling devices to recognize their environment through sound analysis is one of the main objectives of research in computer audition (CA), a broad and challenging area related to computational auditory scene analysis (CASA) [1]. One of the main research areas in CASA is acoustic scene classification (ASC), which attempts to classify digital audio signals into mutually exclusive scene categories. Each acoustic scene is defined as a concept that humans commonly use to identify an acoustic environment containing an ensemble of background noise and sound events in a particular scenario [2]. ASC is an important area of study covering a wide range of applications, including mobile robot navigation [3], audio and security surveillance [4], smart healthcare homes [5], context-aware services, and wild-life monitoring in natural habitats [6].

In the real world, the conditions in which we use the CASA systems will differ from the conditions in which they were developed. Each scene class might contain many types of event sounds and background noises, depending on the location and time. Moreover, the sound can be distorted by non-stationary noise, diverse sound events, and overlapping audio events in the time or frequency domain; it might also be impacted by echoes or reverberant operating conditions. These various factors may exist simultaneously, which increases the complexity of ASC in a combinatorial way [7] and may lead to changes in acoustic data distributions over time. This phenomenon where data distribution evolves or changes over time is called a concept drift [8]. The concept drifts cause predictive models to become less accurate [9]. Simultaneously, the majority of sound recognition solutions assume that data come in a sufficient amount and with a representative and fixed underlying distribution [10]. So in this situation, the ability to detect and adapt the model upon the concept drift is crucial to maintain model performance.

One solution to maintain a good performance of models in non-stationary environments is to retrain and redeploy them periodically. However, this process can be time-consuming and expensive, while selecting the frequency of updates is also not a straightforward task. Another promising approach is using an incremental or evolving learning paradigm [10][11], where a model is iteratively updated upon each newly arrived subset of data [12]. Each iteration is considered as an incremental clustering step towards updating the current model, with every step only processing the newly incoming subset and obtaining data concepts based on the data in this subset.

In this paper, we propose a combine-merge Gaussian mixture model (CMGMM) deployed in an incremental audio scene classifier as an adaptive method to solve the concept drift problem. Gaussian Mixture Model (GMM) is used because it is a simple and efficient approach to model the hidden concept in accoustic scene sounds. Several evolving GMM-based algorithms have been proposed previously. For example, Kristan et al. [13] proposed a clustering algorithm based on an incremental GMM (IGMM) that performs incremental density approximation from observed samples, while Engel and Heinen [14] proposed an IGMM algorithm based on Robbins–Monro stochastic approximation. Furthermore, Song and Wang [15] estimated IGMMs using the expectation-maximization algorithm and proposed a cluster merging strategy based on multivariate statistical tests on the equality of the component's covariance and mean.

However, these recent studies use a passive approach, where adaptation to new concepts is performed periodically. In the proposed method, we employ an active approach, where model adaptation is performed only when a concept drift is detected. Also, the CMGMM considers previous data distributions providing that basic scene classes do not change drastically over time. For example, beach scenes contain typical beaches event sounds like wind and waves. When the scene sounds change, such as the appearance of new seagulls sounds, it will only change some of the patterns of the beach scene sound. However, some typical event sound of the beach scene remains.

The rest of this paper is organized as follows. Section II sets the problem definition, while Section III describes the proposed CMGMM, along with its fundamental equations and algorithm. Section IV outlines the experimental setup, while Section V presents the experimental results. Finally, Section VI presents conclusions and directions for future research.

II. Problem Definition

Consider a dataset $\mathfrak{D}_{init} = \{x_i, y_i\}$, $i = 1{:}n$, where n denotes the number of data points, x_i denotes a scene sound, and y_i denotes the label of x_i . Every x_i contains event sounds and background noise $\hat{x}$ that determine the relation between x_i and y_i; $x_i \in (\hat{x}_1^i, \hat{x}_2^i, .., \hat{x}_a^i)$, where a denotes the number of $\hat{x}$ in x_i. C_{init} represents all $\hat{x}$ in $\mathfrak{D}_{init}$, i.e., $\mathrm{C}_{init} \in (\hat{x}_1^1, \hat{x}_2^1, .., \hat{x}_{a1}^1, \hat{x}_1^2, \hat{x}_2^2, .., \hat{x}_{a2}^2, .., \hat{x}_1^n, , \hat{x}_2^n .., \hat{x}_{aq}^n)$, where q denotes the total number of sound events in $\mathfrak{D}_{init}$. In other words, x_i is a subset of C_{init}, $x_i \subset \mathrm{C}_{init}$.

In this study, we define a concept drift in a scene sound as a change in the hidden relationship defined by a set of $\hat{x}$ that determines the scene label. In the test data $\mathfrak{D}_{test}$, x_i contains a new $\hat{x}$ that changes the relationship between x_i and y_i; $|\mathrm{C}_{init}| < |\mathrm{C}_{test}|$.

III. Combine-Merge Gaussian Mixture Model

This section describes the proposed CMGMM-based algorithm. This algorithm is developed based on a GMM defined by a set of parameters π_j, μ_j, and Σ_j denoting the prior probability, distribution means, and covariance matrix, respectively.

Let a model $\mathcal{M}$ be optimally trained using some initial training data. If during its operation the system detects changes in the likelihood distribution of the incoming data, it will trigger the model adaptation process. The adaptation process begins by creating an optimal model $\mathcal{M}''$ for the newly coming data. This new model aims to respond to new concepts, or concept changes that occurred in the incoming data. Then, the adaptation algorithm combines and merges the components of $\mathcal{M}$ and $\mathcal{M}''$(Figure 1).

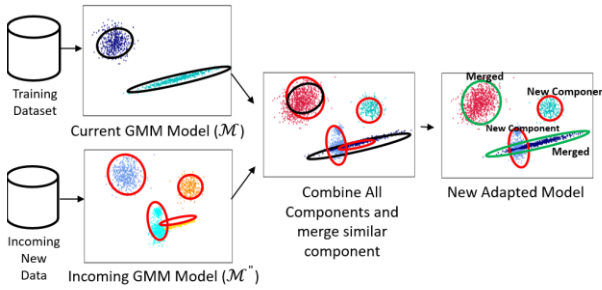
Training Dataset
Current GMM Model ($\mathcal{M}$)
Incoming New Data
Incoming GMM Model ($\mathcal{M}''$)
Combine All Components and merge similar component
Merged
New Component
New Component
Merged
New Adapted Model

Figure 1. Process of the proposed CMGMM-based algorithm

The purpose of the combining process is to add the distributions of any new concepts that may have not been included in the data that model $\mathcal{M}$ was initially trained with, while that of the merging process is to adapt the $\mathcal{M}$ distribution to the new coming data. Therefore, the proposed algorithm makes it possible to deal with both new and old concepts at the same time. The two processes are detailed Section III-D.

A. Feature Extraction

Mel-frequency cepstral coefficients (MFCCs) commonly used as acoustic features [16] are employed in this study. Each MFCC is normalized as

$$Z = \frac{(x-\mu)}{S}, \tag{1}$$

where μ and S denote the mean and standard deviation of the training samples, respectively.

B. Learning Process

The proposed algorithm is first used to build a set of models from the initial dataset $\mathfrak{D}_{init}$ containing training data $x = \{x_1, x_2, \ldots, x_n\}$, where x_n denotes MFCC data. The models are trained Q times using the expectation-maximization (EM) algorithm. For each training cycle, a different K number of components ranging from $K_{\min}$ to $K_{\max}$ are used, where $Q = \mathrm{K}_{\max} - \mathrm{K}_{\min}$. As a result, a set of models $\mathcal{M} = \{\mathcal{M}_1, \mathcal{M}_2, \mathcal{M}_3, .., \mathcal{M}_Q\}$ is obtained based on the different numbers of components.

The next step is to select the best model from $\mathcal{M}$ using the Bayesian information criterion (BIC). In [17], the BIC value of a model $\mathcal{M}_K$ trained over the data set X with K components, BIC(X, M_K), is defined as follows:

$$BIC(X, M_k) \equiv -2 \log L(X, M_K) + v \log N , \tag{2}$$

where L denotes the model likelihood, v denotes the degree of freedom of the parameters in the model, and N denotes the number of training data points. The smaller the BIC value is, the better the model is. In other words, the model with the lowest BIC value is selected. Minimizing the BIC value is the same as maximizing the log-likelihood if both v and N are known [12]. Moreover, maximizing the likelihood is exactly the objective of the classical EM algorithm. Algorithm 1 details the steps of the learning process.

Algorithm 1: Train Best GMM

```
Input  : Initial Dataset $D_{init}$, Minimum Number of
         Component $K_{min}$, and Maximum Number of
         Component $K_{max}$
Output : Best GMM Model
1  $BIC_{best} = \infty$
2  for k= $K_{min}$_component to $K_{max}$
3      $G_{candidate}$= EmTrain($D_{init}$, k)
4      $BIC_{candidate}$ = ComputeBIC ($D_{init}$ , $G_{candidate}$)
5      if $BIC_{candidate} < BIC_{best}$:
6          $G_{best} = G_{candidate}$
7  return $G_{best}$
```

C. Drift Detection Algorithm

In addition to the CMGMM, we also propose a method for detecting concept drifts called kernel density drift detection (KD3). This algorithm works based on the kernel density estimation (KDE) method [18] to estimate the probability density function with one random variable. By comparing two density functions, it is possible to establish the degree of variation in values between the corresponding variables—the greater the variation, the more evidence for the concept drifts. In addition to detecting concept drifts, KD3 also acts as a data collector for the adaptation process by marking a warning zone where data begin to show symptoms of a concept drift.

KD3 requires four input parameters: z, α, β, and h. z denotes a set of likelihood windows, $z = \{z_1, z_2, z_3, \ldots, z_c\}$, where z_c is the current likelihood window that contains a sequence of log-likelihood ℓ from the model prediction, z_c= $\{\ell_1, \ell_2, \ell_3, \ldots, \ell_h\}$. h denotes the window length, α denotes the margin for detecting a concept drift, and β denotes the margin for accumulating the density distance.

KD3 works by comparing two windows, namely, the current window z_c and the previous window z_{c-1}. Let ℓ_n be the latest generated ℓ, and z_c contain h latest ℓ from ℓ_n, $z_c \in [\ell_{n-h}, \ell_n]$ and z_{c-1} contain h latest ℓ from ℓ_{n-h}, $z_{c-1} \in [\ell_{n-2h}, \ell_{n-h}]$. The first step of the algorithm is to compute the Gaussian kernel density function (f_{kde}) of z_c and z_{c-1} . To detect a concept drift, the distance $\grave{d}_t$ between f_{kde} of z_c and z_{c-1} is computed using Eq. 3 within the bounds of $b1$ and $b2$. The bounds are computed based on the maximum and minimum values of the joined ℓ of z_c and z_{c-1}.

$$\grave{d}_t = \frac{1}{2} \int_{b1}^{b2} |f_{kde}(z_1) - f_{kde}(z_2)| \, dz \tag{3}$$

$$z_c \in [\ell_{n-h}, \ell_n],\ z_{c-1} \in [\ell_{n-2h}, \ell_{n-h}],$$

$$b1 = \min(\ell_{n-2h}, \ell_n)\,, b2 = \max\,(\ell_{n-2h}, \ell_n).$$

Finally, the algorithm compares $\grave{d}_t$ with α and β. When $\grave{d}_t > \beta$, the accumulative distance is updated, and if the accumulative distance is equal of greater than α, then the algorithm sends the collected data to the model. Figure 2 and Algorithm 2 illustrate the detailed process of KD3.

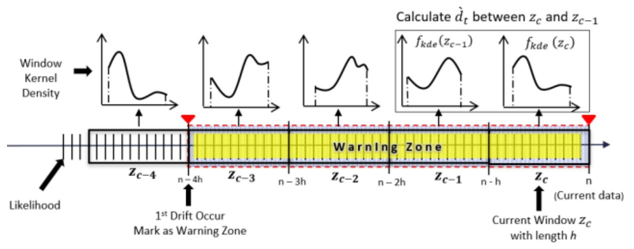

Figure 2. Kernel density drift detection

Algorithm 2: Detect Concept Drift

```
Input   : Set of t likelihood $\ell$, window length $h$, drift
          margin α, warning margin β (α > β)
Output  : Drift Signal, Drifted Dataset
1     Window1 = $\ell[c - h: c]$
2     Window2 = $\ell[c - 2h: c - h]$
3     Boundmin, Boundmax = calcBound($\ell[c - 2h: c]$)
4     KDE1  = estimateKDE(Window1)
5     KDE2  = estimateKDE(Window2)
6     Smpl = GenerateRandomSample(Boundmin, Boundmax)
7     diff = probabilityDistance(KDE1, KDE2 ,Boundmin,
      Boundmax)
8     If (diff ≥ α)
9         resetWarningZoneData ()
10        sendDriftDataToModel ($[c - h: c]$)
11        Return true
12    If (diff ≥ β)
13        accumulativeWarning += diff
14        setWarningZoneData($[c - h: c]$)
15        If(accumulativeWarning ≥ α)
16            resetWarningZoneData()
17            sendWarningZoneDataToModel ()
18            Return true
19    Return False
```

D. Drift Adaptation

We propose the CMGMM as the adaptation method for finding the best model $\mathcal{M}_{best}$ based on three mechanisms, namely, combining Gaussian components, calculating the pairwise similarity between the components, and merging the least similar components. The goal is to find an adapted weighted mixture component that respects the original mixture representing the old concepts. Algorithm 3 details the proposed process of drift adaptation.

The model adaptation process begins by creating a new model $\mathcal{M}_{\text{drift}}$ from data drifts D_{drift} using Algorithm 1, and then combining the existing model $\mathcal{M}_{\text{curr}}$ to form a new model $\mathcal{M}_{\text{all}}$. $\mathcal{M}_{\text{all}}$ contains all components from both $\mathcal{M}_{\text{curr}}$ and $\mathcal{M}_{\text{drift}}$. The next step is to calculate the pairwise distance between components in $\mathcal{M}_{\text{all}}$ using Kullback–Leibler (KL) discrimination presented in [19]. The KL discrimination formula (Eq. 7) enables us to put an upper bound on the discrimination of the mixture before and after the merging process. According to this formula the following components are selected for merging: components with low weights, components with means being close to their variances, and components with similar covariance matrices. When two components are merged, the moment-preserving merging method [20] is used to preserve the mean and covariance of the overall mixture(Eqs. 4-6).

$$w_{ij} = w_i + w_j \tag{4}$$

$$\mu_{ij} = \frac{w_i}{w_{ij}}\mu_i + \frac{w_j}{w_{ij}}\mu_j \tag{5}$$

$$\Sigma_{ij} = \frac{w_i}{w_{ij}}\Sigma_i + \frac{w_j}{w_{ij}}\Sigma_j + \frac{w_i w_j}{{w_{ij}}^2}(\mu_i - \mu_j)(\mu_i - \mu_j)^T \tag{6}$$

$$\begin{aligned} d_{kl}\Big((w_i, \mu_i, \Sigma_i), (w_j, \mu_j, \Sigma_j)\Big) \quad & \quad (7) \\ = \frac{1}{2}\ [w_{ij} \log(\det(\Sigma_{ij})) & \\ - w_i \log(\det\,(\Sigma_i)) & \\ - w_j \log(\det\,(\Sigma_j))] & \end{aligned}$$

Finally, the algorithm iteratively merges the components in $\mathcal{M}_{all}$ ranging from n_{CompMin} to n_{CompMax} to obtain the best model $\mathcal{M}_{\text{best}}$, where n_{CompMin} and n_{CompMax} are the smallest and largest numbers of components derived from $\mathcal{M}_{\text{curr}}$ and $\mathcal{M}_{\text{all}}$, respectively, to be included into $\mathcal{M}_{\text{best}}$. For each iteration, the algorithm finds the least similar components. The merging process follows the rules set by Eqs. 4-6.

Algorithm 3: Combine and Merge Component Adaptation

```
Input   : Trained GMM Model $\mathcal{M}_{curr}$, Drifted Data $D_{drift}$
Output  : Best Adapted GMM Model
1    $\mathcal{M}_{drift}$ = findBestGMM($D_{drift}$)
2    $\mathcal{M}_{all}$  = combineGMMComponent($\mathcal{M}_{curr}$, $G_{drift}$)
3    dist= buildKLDistanceMatrix($\mathcal{M}_{all}$ , target_comp)
4    $n_{CompMin}$ = $\mathcal{M}_{curr}$.n_component
5    $n_{CompMax}$ = $\mathcal{M}_{all}$.n_component
6    for target= $n_{CompMin}$ to $n_{CompMax}$
7      while target > $\mathcal{M}_{all}$.n_component:
8        $i_{min}$,$j_{min}$ = findLeastSimilarComp(dist)
9        $\mathcal{M}_{merge}$[target]=mergeComponent($i_{min}$,$j_{min}$, $\mathcal{M}_{all}$)
10       CalculateAccumulativeBIC($\mathcal{M}_{merge}$, $D_{drift}$)
12   $\mathcal{M}_{best}$ = findLeastAccumulativeBIC($\mathcal{M}_{merge}$)
13   return $\mathcal{M}_{best}$
```

As a result, the reduction process generates a set of merged models $\mathcal{M}_{\text{merge}}$. To select the best $\mathcal{M}_{\text{merge}}$ model, the accumulative BIC is computed by combining sampling data from $\mathcal{M}_{\text{curr}}$, and D_{drift} then computes the BIC value using Eq. 2. The smaller the value of the accumulative BIC is, the better the new adapted model is. Algorithm 3 details the steps of the proposed CMGMM-based method.

IV. Experiments

A. Datasets

The dataset for training and evaluating the CMGMM was generated by selecting 15 types of scenes from the TUT Acoustic Scenes 2017 dataset [21] and TAU Urban Acoustic Scenes 2019 dataset [22]. The overall dataset consisted of 12,000 audio segments of ten seconds each, equally distributed between 15 different scenes and annotated with their ground-truth labels. This dataset was split into a training dataset containing 9000 segments and a test dataset containing 3000 segments. The samples were selected randomly to be included in the two sets. The 15 scenes were beach, bus, cafe/restaurant, car, city center, forest path, grocery store, home, library, metro station, office, park, residential area, train, and tram.

An artificially generated drifted dataset containing three concept drift scenarios, namely, T1, T2, and T3, was used to test the ability of the proposed method to detect and adapt to various drifts. The number of generated sounds for each scenario was 12,000 segments. The test dataset was added to the drifted dataset making the total number of scene segments to be 15,000 for each scenario.

The concept drifts for the three scenarios were simulated by overlaying novel event sounds to the scene sound of each scenario (Figure 3). The novel event sounds were taken from the BBC Sound Class Library [23], and UrbanSound and UrbanSound8K datasets [24] using random positions and gains. In total, 46 classes and 371 new event sounds were added to the scenes of the three scenarios as follows (Figure 3).

- **T1 Scenario**. In scenario T1, several types of unique event sounds related to each scene were overlaid several times with random timing and gains. For example, for the beach scene, sounds representing a dog barking, people talking, people swimming, and raining were overlaid with random timing and gain. None of these sounds were included in the initial training dataset. As a result, the prior distribution of the test data changed, representing the added concept drift.
- **T2 Scenario**. In scenario T2, several sounds were randomly selected from a particular group of event sounds, making the concept drift of this scenario to be more complicated than that of scenario T1. For example, several types of dog-barking sounds from the dog group were added to the beach scene. Similar to scenario T1, the included sounds were related to the respective scene and were overlaid using random timing and gain; none of the added sounds were included in the initial training dataset.
- **T3 Scenario**. Scenario T3 was quite similar to scenario T2. The difference was that a group of event sounds could exist in several scenes (co-existing event classes between scenes). For example, the group of dog-barking sounds could exist in three scenes, e.g., the beach, forest path, and city center scenes. As a result, the prior and posterior distribution of the test data changed.

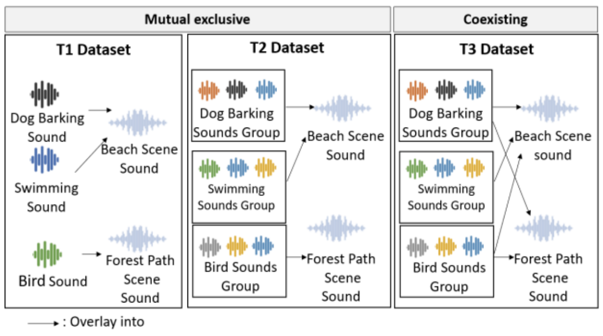

Figure 3. I llustration of the three test scenarios

B. Evaluation Metrics

To evaluate the effectiveness of the proposed method, the following metric was considered:

- Accuracy – the ratio of the correctly predicted observation to the total number of observations.
- F1 score – the weighted average of precision and recall.
- Execution time – the total time of completing the task.

C. Experimental Setup

The effectiveness of the proposed CMGMM was demonstrated by comparing it to the IGMM [25]. The models were tested under the earlier introduced three scenarios (T1, T2, and T3) and the following two approaches, namely, active and passive.

- **Active approach**. This approach assumes that the models detect concept drifts using a concept drift detector and make adaptations upon the detected drifts. In this study, we compared the proposed drift detection method KD3 to the adaptive windowing (ADWIN) method [26] and drift detection method based on the Hoeffding's inequality (HDDM) [27].
- **Passive approach**. This approach assumes that the models adapt continuously as soon as new incoming data are received. Several adaption cycles were tested, namely, 50,100, 150, and 200.

D. Hyperparameters

To evaluate the proposed algorithm, the following best hyperparameter values were selected based on the grid-search results over the test data.

- MFCC hyperparameters: The number of MFCCs was set to 13, the length of the fast Fourier transform (FFT) window was set to 2048, and the number of Mel bands was set to 128.
- KD3 hyperparameters: The window length h was set to 45, the drift margin α was set to 0.001, and the warning margin β was set to 0.00001.
- ADWIN hyperparameters: The delta parameter for the ADWIN was set to 0.002.
- HDDM hyperparameters: The drift confidence level was set to 0.001, and the warning confidence level was set to 0.005.

- CMGMM hyperparameter: Number of components to train the best model parameters ranged from 3 to 30 components.
- IGMM hyperparameters: The minimum and maximum number of components were set to 3 and 60, respectively.

V. Experimental Results

Table 1 lists the experimental results for both active and passive approaches. For the active approach, results are shown for all combinations of the detector and adaptation methods. For the passive approach, the results are shown for different numbers of cycles of the adaptation process.

It can be noticed from Table 1 that better scores were achieved under the active approach. KD3 demonstrated a better performance than ADWIN except for T1. This is mainly because KD3 takes the entire data distribution into account, while ADWIN uses only its average. The drawback of KD3 is its high computational cost. HDDM demonstrated the worst performance among the tested drift detectors. This is because HDDM detected drifts even when they actually did not occur.

In terms of the overall accuracy, CMGMM demonstrated a better performance than IGMM except for T1. Different from CMGMM, IGMM has the parameter of the maximum number of Gaussian components for the distribution of the training data. Therefore, when a concept drift is simple as in T1, IGMM showed a better performance than CMGMM. However, when the concept drift occurred with a complicated combination of event sounds, CMGMM outperformed IGMM because CMGMM allows flexible adaptation using the combining and merging mechanisms and has no limit for the number of Gaussian components.

The combination of CMGMM and KD3 achieved the best performance in the experiment. Figure 4 shows the performance changes of this combination according to time for T1, T2, and T3 scenarios. It can be noticed from the figure that accuracy almost always improves in scenario T2 after the detection of the concept drift. In T3, accuracy is not improved from around 6,000th to 12,000th data point because there are a lot of deeper decrement compared to T1 and T2, as T3 simulates a more complicated distribution of the concept drift. In T3 the possibility of newly coming data have different concept with adapted model is higher than T1 and T2. While the performance is the best in T1, it seems that concept drifts are detected more often than they actually occurred.

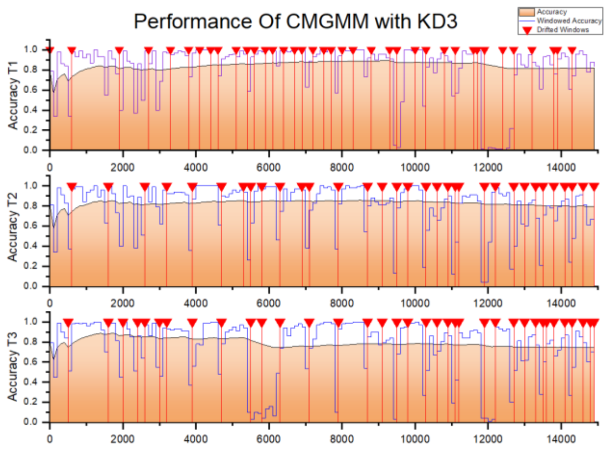

Figure 4. Overall and windowed performance of CMGMM and KD3 in the three scenarios:T1, T2, and T3.

Figure 5 shows the change in performance of CMGMM with ADWIN overtime for T1, T2, and T3 scenarios. In scenario T1, the performance of this combination is good, although the number of adaptations is reduced compared to the combination of CMGMM and KD3 in the same scenario. However, in the other two scenarios featuring complicated concept drifts, ADWIN demonstrates delays in detecting the concept drifts at several points. These delays in detection lead to delays in adaptation, causing the overall performance of CMGMM with ADWIN to decrease over time from around 90% to 70%. For example, in scenario T2, there is a significant decrease in performance from 3000th to 4000th data points. Hence, both the accuracy of drift detection and speed of adaptation to drifts are vital aspects impacting the overall model performance.

TABLE I. Experiment Result in Active and Passive Scenario

ACTIVE APPROACH											
ADAPTOR	DETECTOR	ACCURACY			F1 SCORE			EXECUTION TIME			Drift
		T1	T2	T3	T1	T2	T2	T1	T2	T3	
CMGMM*	KD3*	0.8373	**0.7962**	**0.7409**	0.8432	**0.7993**	**0.7460**	128.06	115.07	110.49	39
	ADWIN	**0.8401**	0.6415	0.6332	**0.8418**	0.6379	0.6379	83.07	84.22	85.44	23
	HDDM	0.2762	0.2627	0.2990	0.3184	0.2992	0.3406	84.81	84.53	83.11	373
IGMM	KD3*	0.8283	**0.7574**	**0.6622**	0.8173	**0.7499**	**0.6488**	120.04	128.75	120.50	35
	ADWIN	**0.8419**	0.5711	0.6057	**0.8423**	0.5722	0.6063	82.80	84.219	83.08	21
	HDDM	0.2363	0.2507	0.2032	0.2436	0.3055	0.2675	84.37	87.55	84.87	350
PASSIVE APPROACH											
ADAPTOR	ADAPTATI ON CYCLE	ACCURACY			F1 SCORE			EXECUTION TIME			Drift
		T1	T2	T3	T1	T2	T3	T1	T2	T3	
CMGMM*	50	0.5621	0.4451	0.4003	0.5719	0.4615	0.4373	83.178	82.945	83.163	-
	100	0.7424	0.6547	0.6434	0.7451	0.6583	0.6476	83.688	82.265	83.704	-
	150	0.8002	**0.7437**	**0.7301**	0.8043	**0.7482**	**0.7327**	84.527	82.298	89.555	-
	200	0.7602	0.7073	0.6904	0.7663	0.714	0.7001	83.414	85.922	84.594	-
IGMM	50	0.5365	0.4209	0.3687	0.5458	0.4319	0.3888	84.467	82.776	82.655	-
	100	0.7324	0.643	0.6371	0.736	0.6448	0.6388	82.693	82.652	82.199	-
	150	**0.8056**	0.7401	0.7291	**0.8107**	0.7431	0.7223	83.659	82.645	83.453	-
	200	0.7528	0.7149	0.6899	0.7609	0.722	0.6981	84.597	84.228	85.797	-

(*) Proposed Method

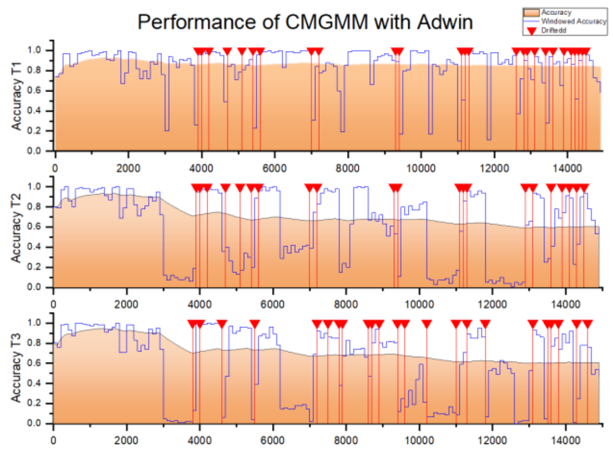

Figure 5. Overall and windowed performance of CMGMM and ADWIN in the three scenarios:T1, T2, and T3.

Furthermore, the adaptation cycle (i.e., the number of data points to update) plays an important role in achieving a good performance in the passive approach. While IGMM shows its best performance in T1 and CMGMM – in T2 and T3, both models have the same optimal adaptation cycle of 150 data points. This suggests the importance of finding the best adaptation cycle, which may vary from one dataset to another. For example, when the adaptation cycle for CMGMM in T1 increased from 50 to 100 and 150, the accuracy score increased from 0.5621 to 0.7424 and 0.8002, respectively. However, when the adaptation cycle further increased to 200 or more, the accuracy score decreased to 0.7602.

VI. Conclusion

This paper presented the KD3 algorithm for detecting concept drifts based on model log-likelihood and CMGMM-based algorithm for adapting to the detected concept drifts. The experimental results demonstrated that the proposed algorithms work well in detecting and adapting to three types of drift scenarios. By comparing the active and passive approaches to detecting concept drifts, the number of data to adapt to the model is a crucial factor impacting the method's accuracy. As part of our future work, we plan to improve the proposed KD3 concept drift detection algorithm to optimally choose the number of data points to consider when detecting a drift and decrese it computation time.

Acknowledgment

This work was supported by JSPS KAKENHI Grant Number JP20K12079

References

[1] D. Wang and G. J. Brown, "Computational auditory scene analysis: Principles, algorithms, and applications," *Comput. Audit. Scene Anal. Princ. Algorithms, Appl.*, pp. 1–395, 2006.

[2] M. Valenti, S. Squartini, A. Diment, G. Parascandolo, and T. Virtanen, "A convolutional neural network approach for acoustic scene classification," *Proc. Int. Jt. Conf. Neural Networks*, vol. 2017-May, pp. 1547–1554, 2017.

[3] S. Chu, S. Narayanan, C. C. Jay Kuo, and M. J. Matarić, "Where am I? Scene recognition for mobile robots using audio features," *2006 IEEE Int. Conf. Multimed. Expo, ICME 2006 - Proc.*, vol. 2006, no. July, pp. 885–888, 2006.

[4] S. Chandrakala and S. L. Jayalakshmi, "Environmental Audio Scene and Sound Event Recognition for Autonomous Surveillance," *ACM Comput. Surv.*, vol. 52, no. 3, pp. 1–34, 2019.

[5] M. Vacher, A. Fleury, F. Portet, J.-F. Serignat, and N. Noury, "Complete Sound and Speech Recognition System for Health Smart Homes: Application to the Recognition of Activities of Daily Living," *New Dev. Biomed. Eng.*, 2010.

[6] A. J. Eronen *et al.*, "Audio-based context recognition," *IEEE Trans. Audio, Speech Lang. Process.*, vol. 14, no. 1, pp. 321–329, 2006.

[7] S. Ntalampiras, "Automatic analysis of audiostreams in the concept drift environment," *IEEE Int. Work. Mach. Learn. Signal Process. MLSP*, vol. 2016-Novem, pp. 0–5, 2016.

[8] R. Elwell and R. Polikar, "Incremental learning of concept drift in nonstationary environments," *IEEE Trans. Neural Networks*, vol. 22, no. 10, pp. 1517–1531, 2011.

[9] J. Gama, I. Žliobaitė, A. Bifet, M. Pechenizkiy, and A. Bouchachia, "A survey on concept drift adaptation," *ACM Comput. Surv.*, vol. 46, no. 4, pp. 1–37, 2014.

[10] T. R. Hoens, R. Polikar, and N. V. Chawla, "Learning from streaming data with concept drift and imbalance: An overview," *Prog. Artif. Intell.*, vol. 1, no. 1, pp. 89–101, 2012.

[11] I. Žliobaitė, "Learning under Concept Drift: an Overview," pp. 1–36, 2010.

[12] C. Chen *et al.*, "Improving Accuracy of Evolving GMM under GPGPU-Friendly Block-Evolutionary Pattern," *Int. J. Pattern Recognit. Artif. Intell.*, vol. 34, no. 3, pp. 1–34, 2020.

[13] M. Kristan, D. Skocaj, and A. Leonardis, "Incremental learning with Gaussian mixture models," in *13th Computer Vision Winter Workshop (Slovenian Pattern Recognition Society, Moravske Toplice, Slovenia, 2008)*, 2008, pp. 25–32.

[14] P. M. Engel and M. R. Heinen, "Incremental learning of multivariate Gaussian mixture models," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 6404 LNAI, pp. 82–91, 2010.

[15] M. Song and H. Wang, "Highly efficient incremental estimation of Gaussian mixture models for online data stream clustering," *Intell. Comput. Theory Appl. III*, vol. 5803, p. 174, 2005.

[16] P. Pal Singh, "An Approach to Extract Feature using MFCC," *IOSR J. Eng.*, vol. 4, no. 8, pp. 21–25, 2014.

[17] C. Fraley, "How Many Clusters? Which Clustering Method? Answers Via Model-Based Cluster Analysis," *Comput. J.*, vol. 41, no. 8, pp. 578–588, 1998.

[18] E. Parzen, "On the Estimation of Probability Density Functions and Mode," *Ann. Math. Stat.*, vol. 33, pp. 1065–1076, 1962.

[19] A. R. Runnalls, "Kullback-Leibler approach to Gaussian mixture reduction," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 43, no. 3, pp. 989–999, 2007.

[20] J. L. Williams and P. S. Maybeck, "Cost-function-based Gaussian mixture reduction for target tracking," *Proc. 6th Int. Conf. Inf. Fusion*, vol. 2, pp. 1047–1054, 2003.

[21] A. Mesaros, T. Heittola, and T. Virtanen, "TUT Acoustic scenes 2017, Development dataset," Mar. 2017.

[22] T. Heittola, A. Mesaros, and T. Virtanen, "TAU Urban Acoustic Scenes 2019, Development dataset," Mar. 2019.

[23] "BBC Sound Effects." [Online]. Available: http://bbcsfx.acropolis.org.uk/. [Accessed: 20-Mar-2019].

[24] J. Salamon, C. Jacoby, and J. P. Bello, "A dataset and taxonomy for urban sound research," *MM 2014 - Proc. 2014 ACM Conf. Multimed.*, no. November, pp. 1041–1044, 2014.

[25] J. M. Acevedo-Valle, K. Trejo, and C. Angulo, "Multivariate regression with incremental learning of Gaussian mixture models," *Front. Artif. Intell. Appl.*, vol. 300, pp. 196–205, 2017.

[26] A. Bifet and R. Gavaldà, "Learning from time-changing data with adaptive windowing," *Proc. 7th SIAM Int. Conf. Data Min.*, no. April, pp. 443–448, 2007.

[27] I. Frías-Blanco, J. Del Campo-Ávila, G. Ramos-Jiménez, R. Morales-Bueno, A. Ortiz-Díaz, and Y. Caballero-Mota, "Online and non-parametric drift detection methods based on Hoeffding's bounds," *IEEE Trans. Knowl. Data Eng.*, vol. 27, no. 3, pp. 810–823, 2015.